Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О.В. Непомнящий
подпись инициалы, фамилия
« ___» _____ 2018 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника
код и наименование направления

Моделирование сети произвольной топологии и оценка ее производительности средствами Omnet++
тема

Руководитель _____ доцент, канд. техн. наук О.А.Русанова
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник _____ О.Ю. Гаас
подпись, дата инициалы, фамилия

Нормоконтролер _____ В.И. Иванов
подпись, дата инициалы, фамилия

Красноярск 2018

РЕФЕРАТ

Выпускная квалификационная работа по теме «Моделирование работы сети произвольной топологии, и оценка ее производительности средствами Omnet++» содержит 78 страниц, 27 рисунков, 7 таблиц, и список использованных источников из 16 наименований.

СЕТИ ЭВМ, ПРОИЗВОДИТЕЛЬНОСТЬ, OMNET++, МОДЕЛЬ, ПАКЕТ, ОБМЕН ИНФОРМАЦИЕЙ, ПОТОК ДАННЫХ

Цель – разработка методики тестирования сетей ЭВМ произвольной топологии средствами Omnet++ и программного кода на языке NED для реализации топологий сетей.

Задачи:

- рассмотреть существующие методики оценки производительности;
- выполнить обзор инструментов измерения производительности в среде Omnet++
- разработать методику измерения производительности сети используя инструментарий Omnet++
- выполнить моделирование сетей заданной топологии, для определения производительности используя разработанную методику
- провести анализ полученных результатов и оценить эффективность методики.

СОДЕРЖАНИЕ

Введение

Производительность сети (скорость передачи данных прикладного уровня) является важной характеристикой, одной из составляющей понятия «качество обслуживания» [1].

Потенциально высокая производительность – одно из основных преимуществ распределенных систем. Чаще всего при проектировании, настройке и оптимизации сети используются такие показатели, как средняя и максимальная скорость передачи данных [2]. Средняя скорость, с которой передают данные отдельный элемент или сеть в целом, позволяет оценить работу сети на протяжении длительного времени, в течение которого пики и спады интенсивности трафика компенсируют друг друга. Максимальная скорость позволяет оценить, как сеть будет справляться с пиковыми нагрузками, характерными для особых периодов работы. Производительность сети зависит от множества факторов: пропускной способности и загруженности каналов передачи данных, топологии сети, времени прохождения пакетов. На производительность сети влияет оборудование и технологии, используемые в сети.

Так как существует множество факторов, влияющих на скорость передачи данных, оценка производительности сети – не простая задача, поэтому существует множество различных методик её измерения [5]. Например, сервисы или тесты скорости представленные на различных веб-ресурсах могут измерять ширину канала до некоторого сервера. Но при использовании данного метода происходит измерение трассы до определенного сервера, а не конкретного канала связи или участка сети.

Для оценки производительности сети на этапе её проектирования можно применять специализированные инструменты, сетевые симуляторы, например Omnet++[11]. Благодаря моделированию в данной среде, можно выявить и решить проблемы с производительностью, например, выявить «слабые места», а также спланировать дальнейшее развитие сети. В данной

работе представлена методика оценки производительности сети с использованием Omnet++. Предложенная методика опробована в ходе лабораторных экспериментов, а анализ полученных результатов показывает, что применение разработанной методики позволяет адекватно оценивать производительность сети.

1. Анализ предметной области

1.1 Обзор существующих методик измерения производительности сети

Для оценки производительности сети существует ряд методов [7]. Первый метод - ручное тестирование, например, с помощью передачи файла фиксированного размера по FTP. Недостаток данного метода заключается в том, что скорость передачи данных по протоколу FTP зависит от множества факторов, например, RTT (время между отправкой запроса и получением ответа), загруженности процессора и системы хранения данных сервера.

Рассмотрен ряд инструментов для измерения производительности и методик измерений:

1) LAN Speed Test (рисунок 1) [14]. Разработана компанией Totusoft. Программа измеряет производительность сети указанным выше методом (создает файл в памяти, а затем передает его в обоих направлениях по протоколу FTP).

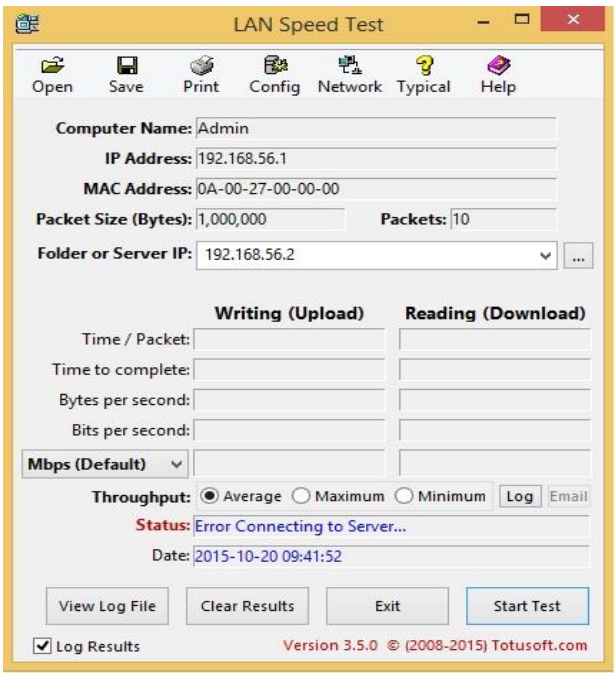

Рисунок 1 – LAN Speed test

2) LAN Bench (рисунок 2) – простое приложение для тестирования производительности сети между двумя компьютерами, основан на Winsock 2.2 [15]. На одном из них программа запускается в режиме «Сервер», и «слушает» входящие подключения, а на другом — запускается в режиме «Клиент». Программа тестирует производительность сети, используя только протокол TCP. Можно настроить длительность теста, размер пакета для отправки в КБ и количество одновременных TCP-соединений, которых не должно превышать 20. Программа затрачивает минимум ресурсов системы и не требует установки.

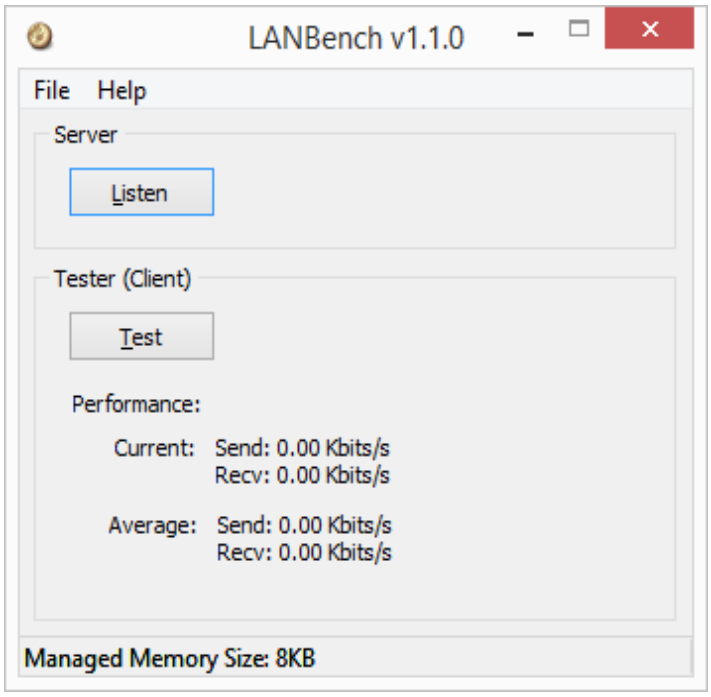

Рисунок 2 – Lan Bench

3) Pathrate и pathload [10]. Пакет pathrate (Constantine Dovrolis) позволяет определять максимальную пропускную способность (bandwidth) и также состоит из 2 утилит:

pathrate_snd («передатчик»), которая слушает команды на 48699/tcp, получив команду тестирования обеспечивает UDP поток на приёмник 48698/udp; "

pathrate_rcv («приёмник») запускается на другом конце, передаёт команду тестирования на «передатчик»;

Особенностью данной программы является способность определять общую пропускную способность канала невзирая на его текущую загрузку. Результат является статистическим и ненадёжным (основывается на задержках пакетов в цепочке пакетов). На этапе инициализации pathrate определяет наличие балансировщиков трафика (измеряется пропускная способность одного канала) и ограничителей трафика (измеряется пиковая скорость), вычисляется размер буферов (определяется максимальной длиной цепочки до потери нескольких пакетов подряд). Если текущий трафик в канале невелик и постоянен, то пропускную способность канала можно определить непосредственно. Полный цикл может занять полчаса. При этом определяется зависимость задержек от размера пакетов и числа пакетов в цепочке.

4) Iperf (рисунок 3) [16]. Генерирует и TCP, и UDP трафик. Измерение осуществляется следующим образом: на одном ПК запускаем iperf в режиме «сервер», на втором в режиме «клиент» с указанием ip-адреса первого ПК («сервера»). Через заданное время показываются результаты измерений. Обилие опций для сервера и клиента позволяет контролировать процесс тестирования до мелочей. Если сервер запускается в интерактивном режиме, то он также выдаёт результаты измерения. При запуске теста между клиентом и сервером устанавливается управляющее соединение (по умолчанию - 5001/tcp), тестовые данные пересылаются по первому доступному «пользовательскому» порту.

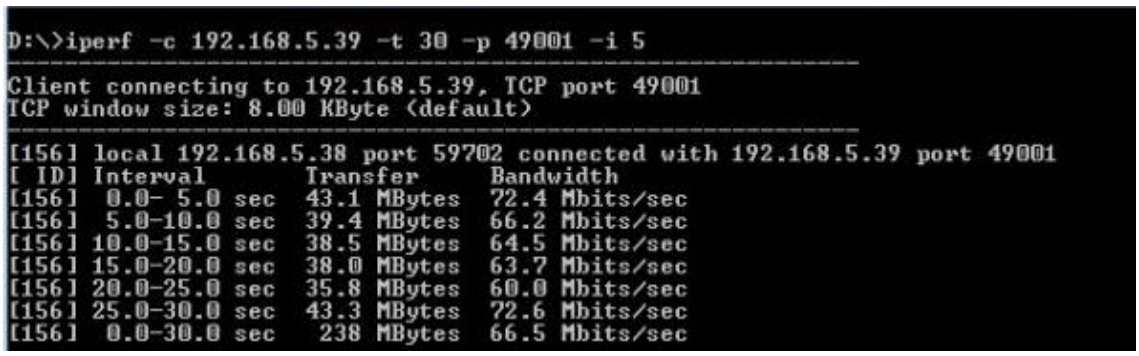

Рисунок 3 – Окно Iperf

Iperf позволяет определить:

- пропускную способность сети при использовании TCP;

- величину окна TCP по умолчанию и оптимальное значение;

- пропускную способность сети в режиме UDP при заданной скорость передачи данных;

- потери UDP пакетов;

- отклонения задержки передачи UDP пакетов от среднего значения (синхронизация времени не требуется);

- производительность сети в режиме групповых рассылок (multicast).

Первые две программы измеряют только скорость передачи данных по протоколу TCP. Но для полноценной оценки производительности сети этого недостаточно. В локальной сети пользователи могут пользоваться не только передачей каких-либо файлов, электронной почтой и т.п., но и приложениями, использующими UDP, например, видеоконференциями. Поэтому необходимо оценивать производительность сети в режиме передачи данных по протоколу UDP.

Iperf или pathrate можно использовать только в существующей сети. Так как стоит задача оценить сеть на стадии её проектирования, необходима методика оценки производительности сети с использованием симуляторов. В данной работе разработана методика оценки производительности с использованием сетевого симулятора Omnet++. В работе будет использоваться метод, аналогичный методу, используемому в Iperf, так как

метод, реализованный в данной программе, хорошо проявил себя в задаче оценки производительности сети, и может быть реализован в данной среде моделирования.

1.2 Обзор среды моделирования Omnet++

Omnet++ - расширяемая, модульная среда, основанная на компонентах библиотек моделирования и C ++, предназначенная для создания сетевых симуляторов. Omnet++ позволяет моделировать проводные и беспроводные сети связи, внутрисхемные сети, сети очередей и т. д. [11]

Разработчик – Андраш Варга (Andras Varga). Это продукт с открытым исходным кодом, является бесплатным только для академического и некоммерческого использования.

OMNeT++ приобрел широкую популярность в качестве платформы сетевого моделирования в научном сообществе, а также в промышленных установках. OMNeT++ предоставляет компонентную архитектуру для моделей. Компоненты (модули) создаются с использованием C++, затем собираются в более крупные компоненты и модели с использованием языка высокого уровня (NED). Данный инструмент имеет расширенную поддержку графического интерфейса пользователя, и благодаря своей модульной архитектуре ядро моделирования (и модели) может быть легко встроено в пользовательские приложения.

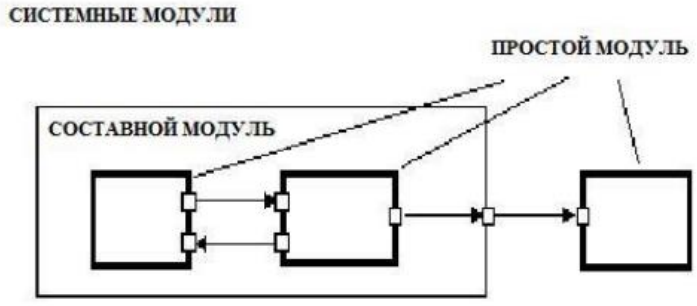

Рисунок 4 – Модули Omnet++

OMNeT++ работает на Windows, Linux, Mac OS X и других UNIX-подобных системах и состоит из следующих компонентов:

- библиотека ядра моделирования;

- язык описания топологии NED (англ. NEtwork Description);

- IDE на базе платформы Eclipse;

- графический интерфейс для выполнения моделирования Tkenv, который ссылается на исполняемый файл симуляции;

- пользовательский интерфейс командной строки для выполнения моделирования (Cmdenv);

- документация, примеры моделей и др.

OMNeT++ предоставляет эффективные инструменты для пользователя, чтобы описать структуру реальной системы.

Модель сети в OMNeT++ состоит из следующих частей:

- описания топологий на языке NED (файлы с расширением .ned), которые описывают структуру модуля с параметрами, шлюзами и т. д. Такие файлы могут быть созданы с помощью любого текстового редактора, но IDE OMNeT++ обеспечивает поддержку двустороннего графического и текстового редактирования (рисунки 5, 6);

```
network Backbone
{
    parameters:
        @display("p=10,10;b=736,568");
    types:
        channel C extends ThruputMeteringChannel
        {
            datarate = 100Mbps;
            thruputDisplayFormat = "#N";
        }
    submodules:
        R1: OSPFRouter {
            parameters:
                @display("p=210,357");
            gates:
                ethg[3];
        }
        N1: EtherHub {
            parameters:
                @display("p=210,272");
            gates:
                ethg[2];
        }
        N2: EtherHub {
            parameters:
                @display("p=122,428");
            gates:
```

Рисунок 5 – Редактор исходного кода NED

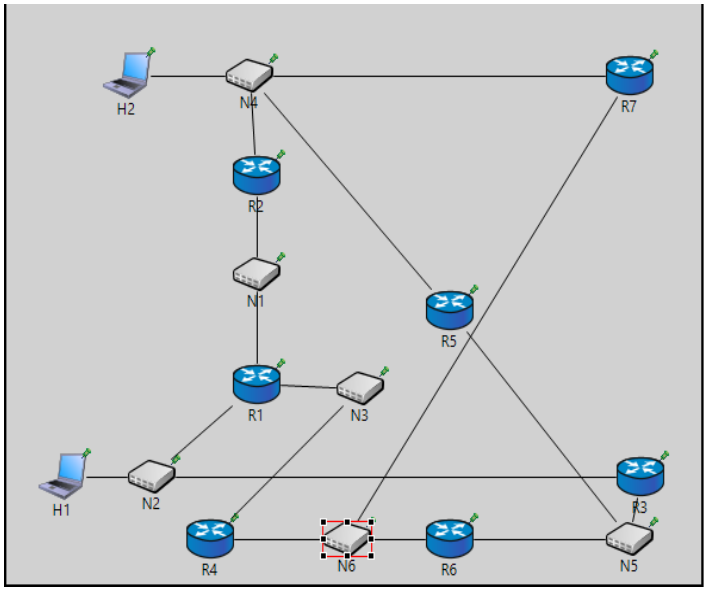

Рисунок 6 – Графический редактор NED

- исходные коды модуля, которые представляют собой C++ файлы с расширением .h или .cc.

- файлы описания параметров автономных систем, сценариев и т.д. (.xml файлы)

- файлы результатов моделирования .anf

NED-файлы загружаются динамически в исходных текстовых формах при запуске программы моделирования. Моделирование выполняется в среде, которая обеспечивается библиотеками пользовательского интерфейса. (Envir, Cmdenv, Tkenv) – среда контролирует процесс ввода данных, вывода результатов, отладки, визуализации и анимации имитационной модели. Программа моделирования может быть скомпилирована как отдельный исполняемый файл так, что она может быть запущена на других компьютерах, где OMNeT++ не установлена.

OMNeT++ может быть расширен с помощью специальных библиотек моделирования. Наиболее известной и распространенной является INET Framework, исходный код которой открыт. INET предоставляет протоколы, агенты и другие модели для исследователей и студентов, работающих с сетями передачи данных. INET особенно полезна при разработке и утверждении новых протоколов, при изучении новых экзотических сценариев. INET содержит модели для стека протоколов Интернета (TCP, UDP, IPv4, IPv6, OSPF, BGP и т. д.), для проводных и беспроводных протоколов канального уровня (Ethernet, PPP, IEEE 802.11 и т. д.), для поддержки мобильности, протоколов MANET (Mobile Ad hoc Network), DiffServ, включает несколько моделей приложений, много других протоколов и компонентов.

1.3 Обзор библиотеки моделирования INET Framework

INET Framework — библиотека моделирования для Omnet++, которая содержит модули, реализующие стек протоколов Интернета (TCP, UDP,

IPv4, IPv6, OSPF, BGP и т. д.), протоколы канального уровня (Ethernet, PPP, IEEE 802.11 и т. д.), и другие. Библиотека также содержит большое количество готовых сетевых моделей.

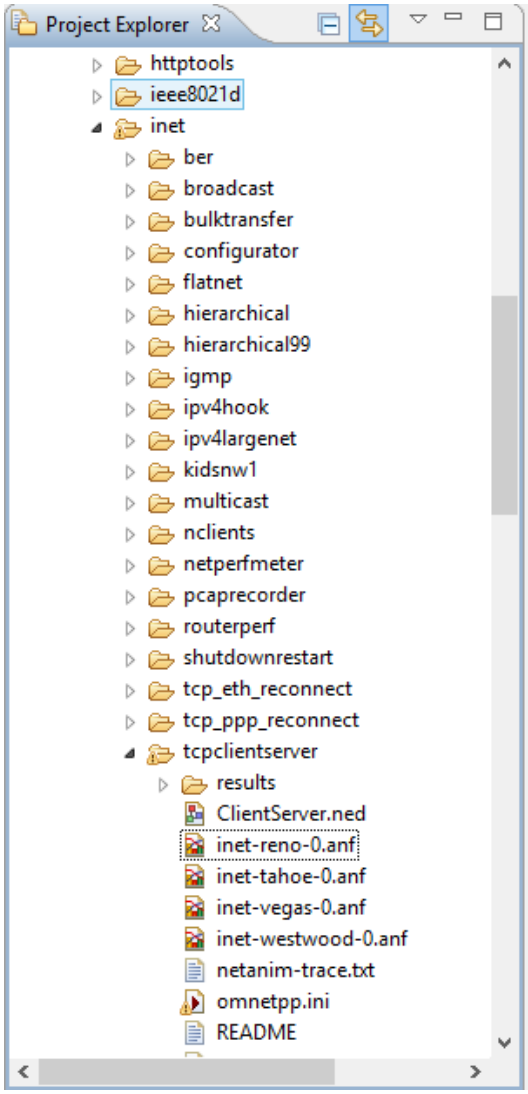

Рисунок 7 – Структура сетевых моделей в состава Inet Framework

Рассмотрим структуру сетевой модели на примере модели tcpclientserver. Модель имеет следующую файловую структуру:

1) ClientServer.ned — файл исходного кода на языке NED, в котором содержится структурная схема сетевой модели.

Сетевая модель строится на основе модулей, идущих в комплекте INET Framework:

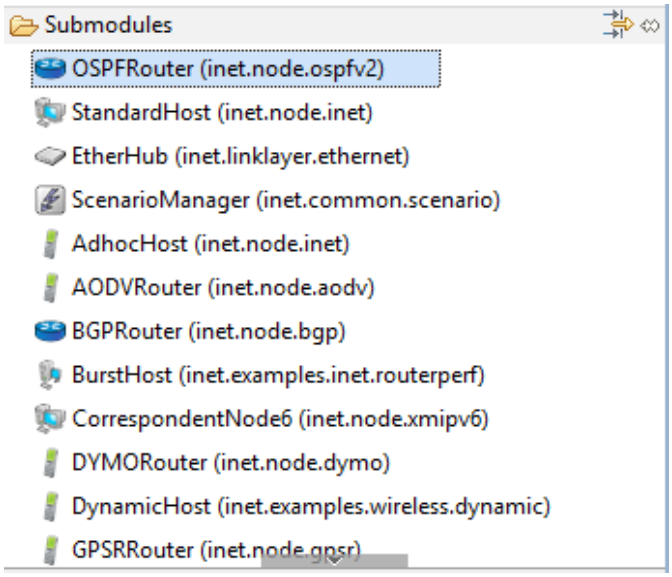

Рисунок 8 – Модули для построения сети

- StandardHost — модель хоста IP
- OSPFRouter — модель маршрутизатора IP с поддержкой OSPF
- EtherHub — модель хаба Ethernet
- EtherSwitch — модель коммутатора Ethernet

Модели хоста и маршрутизатора описываются следующей структурной схемой:

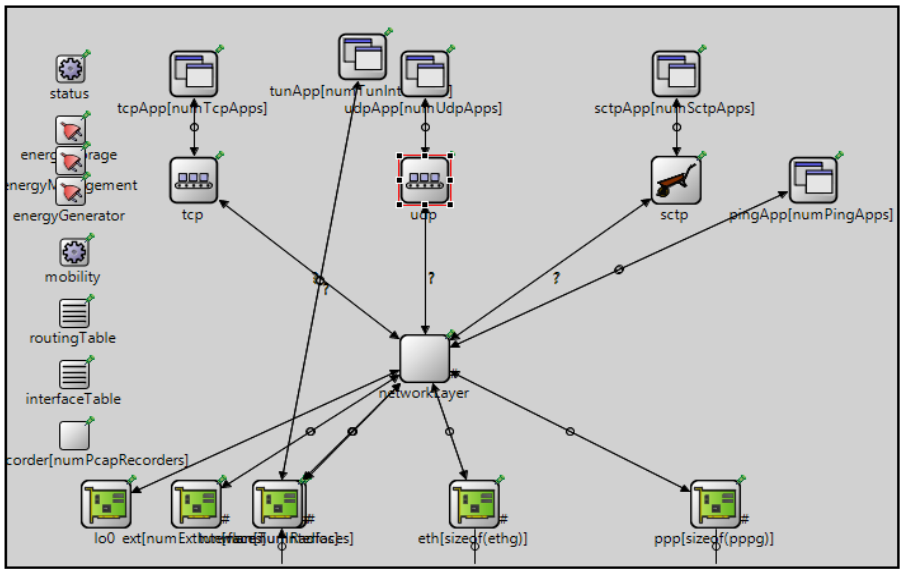

Рисунок 9 – Структурная схема хоста и маршрутизатора

- lo0, eth, ppp — модули, имитирующие работу сетевых интерфейсов
- networkLayer — модуль, реализующий протокол IP
- tcp, udp — модули, реализующие транспортные протоколы
- tcpApp, udpApp, pingApp — модули, реализующие прикладной уровень сети.

Связи между модулями сетевой модели и их параметры задаются в файле .ned следующим образом:

connections:

client1.ethg++ <--> C <--> server.ethg++;

types:

channel C extends DatarateChannel

{

datarate = 100Mbps;

}

2) Файл Omnetpp.ini содержит настройки компонентов сетевой модели. В данном файле можно задать параметры TCP и UDP приложений, MTU сетевых интерфейсов, IP-адреса хостов, и так далее.

```
## tcp apps
**.numTcpApps = 1
**.client*.tcpApp[*].typename = "TCPSessionApp"
**.client*.tcpApp[0].active = true

# udp app
**.numUdpApps = 1
**.client1.udpApp[0].typename = "UDPBasicApp"
**.client1.udpApp[0].destPort = 1234
**.client1.udpApp[0].messageLength = 1024 bytes
**.client1.udpApp[0].sendInterval = 0.00009s
**.client1.udpApp[0].startTime = 10s
**.client1.udpApp[0].stopTime = this.startTime + 10s
**.client1.udpApp[0].destAddresses = "server"
**.server.udpApp[0].typename = "UDPEchoApp"
**.server.udpApp[0].localPort = 1234


**.eth[*].mac.mtu = 9000B
*.configurator.config=xml("<config><interface hosts='*'
```

Рисунок 10 – Файл omnetpp.ini

3) Файл ASConfig.xml содержит опции, управляющие маршрутизацей в сетевой модели (IP-подсети, номера AS, зоны OSPF, и так далее).

```
<?xml version="1.0"?>
<OSPFASConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="OSPF.xsd">

  <!-- Areas -->
  <Area id="0.0.0.0">
    <AddressRange address="R1>N1" mask="R1>N1/" status="Advertise" />
    <AddressRange address="R1>N2" mask="R1>N2/" status="Advertise" />
    <AddressRange address="R1>N3" mask="R1>N3/" status="Advertise" />
    <AddressRange address="R2>N4" mask="R2>N4/" status="Advertise" />
```

Рисунок 11 – Файл ASconfig.xml

4) Файлы результата моделирования *.anf, которые появляются после проведения экспериментов. Информацию из .anf файлов можно представить в виде графиков, диаграмм или средних целых значений.

1.4 Выводы по разделу

На основании выполненного анализа отметим, что существующие инструментальные средства анализа сетей обладают рядом преимуществ, например: адекватность результатов анализа, создание профиля нагрузки сети соответствующему реальным сетевым приложениям. Тем не менее, к недостаткам инструментария можно отнести ограниченность область его применения (инструментарий можно использовать только в уже существующих сетях), а так же высокую стоимость анализа сети. Исключение составляет программный пакет Omnet++, который не имеет указанных выше недостатков. Кроме того программа бесплатна для использования в академических целях и может быть расширена с помощью специальных библиотек моделирования.

В состав пакета входит библиотека моделирования имеющая встроенную поддержку протоколов Интернета (TCP, UDP, IPv4, IPv6, OSPF, BGP и т. д.), а так же поддерживаются протоколы канального уровня

(Ethernet, PPP, IEEE 802.11 и т. д.), и другие. Основным достоинством данного продукта следует считать большое количество готовых сетевых моделей.

2 Методика измерения производительности сети в среде Omnet++

2.1 Этап 1 - Создание модели сети

На первом этапе создается модель сети, для этого выполняется анализ заданной топологии и выбираются компоненты INET Framework. Выбор компонент осуществляется в соответствии с таблицей 1.

Таблица 1 – Компоненты INET Framework

Элемент сети	Компонент Omnet++
Коммутатор Ethernet	EtherSwitch
Коммутатор MPLS	RSVP_LSR, LDP_LSR
Маршрутизатор IP	OSPFRouter, BGPRouter, и.т.д.
Хаб Ethernet	EtherHub
Точка доступа 802.11	AccessPoint
Хост IP	StandardHost
Хост Ethernet	EtherHost
Хост 802.11	WirelessHost
Сеть Ethernet шинной топологии	EtherBus
Сеть IP произвольной топологии	InternetCloud
Радиоканал передачи данных	RadioMedium
Канал топологии точка-точка	Eth*, DatarateChannel, DelayChannel, fiberline

Далее, в зависимости от топологии сети, определяются каналы связи и выполняется соединение компонент. Способ описания канала связи представлен на рисунке 5. Каждый тип каналов связи описывается в исходном коде следующим образом:

channel <тип канала> extends <название компонента>

{

disabled=<true/false>

datarate=<пропускная способность>

delay=<задержка передачи данных>

ber=<процент битовых ошибок>

per=<процент потерянных пакетов>

}

Для соединения допускается использование графического редактора. Следует учесть, что при использовании графического редактора существует вероятность ошибки. Так если у компонента должно быть несколько соединений, то в графическом редакторе программа позволяет присоединять канал связи только к одному и тому же порту (например, eth[0]). Поэтому, рекомендуется для сложных конфигураций все соединения компонентов лучше задавать в NED-коде (рисунок 12).

```
connections:
    R1.ethg[0] <--> C <--> N1.ethg[1];
    R1.ethg[1] <--> C <--> N2.ethg[2];
    N1.ethg[0] <--> C <--> R2.ethg[1];
    R2.ethg[0] <--> C <--> N4.ethg[2];
    N4.ethg[1] <--> C <--> R5.ethg[1];
    R5.ethg[0] <--> C <--> N5.ethg[2];
    N2.ethg[0] <--> C <--> R3.ethg[1];
    R3.ethg[0] <--> C <--> N5.ethg[1];
    N5.ethg[0] <--> C <--> R6.ethg[0];
```

Рисунок 12 – Соединение компонентов сети

Кроме каналов связи, необходимо выполнить настройку параметров сетевого уровня (IP-адреса, таблицы маршрутизации, и.т.д.).

Для настройки IP-адресов элементов сети (хосты и маршрутизаторы) используется конфигуратор IPv4NetworkConfigurator в котором задаются адрес и маска для каждого элемента (рисунок 13).

```
configurator: IPv4NetworkConfigurator {
    parameters:
        config = xml("<config>"+
                    "<interface hosts='R2' towards='N1 R1' address='192.168.1.1' netmask='255.255.255.0' />"+
                    "<interface hosts='R1' towards='N1 R2' address='192.168.1.2' netmask='255.255.255.0' />"+

                    "<interface hosts='R3' towards='N2 R1' address='192.168.2.1' netmask='255.255.255.0'  />"+
                    "<interface hosts='H1' address='192.168.2.2' netmask='255.255.255.0' />"+
                    "<interface hosts='R1' towards='N2 R3' address='192.168.2.3' netmask='255.255.255.0' />"+

                    "<interface hosts='R1' towards='N3 R4' address='192.168.3.1' netmask='255.255.255.0'  />"+
                    "<interface hosts='R4' towards='N3 R1' address='192.168.3.2' netmask='255.255.255.0'  />"+

                    "<interface hosts='H2' address='192.168.4.1' netmask='255.255.255.0' />"+
                    "<interface hosts='R5' towards='N4 H2' address='192.168.4.2' netmask='255.255.255.0'  />"+
                    "<interface hosts='R2' towards='N4 H2' address='192.168.4.3' netmask='255.255.255.0'  />"+
                    "<interface hosts='R7' towards='N4 H2' address='192.168.4.4' netmask='255.255.255.0'  />"+

                    "<interface hosts='R6' towards='N5 R5' address='192.168.5.1' netmask='255.255.255.0' />"+
                    "<interface hosts='R3' towards='N5 R5' address='192.168.5.2' netmask='255.255.255.0'  />"+
                    "<interface hosts='R5' towards='N5 R6' address='192.168.5.3' netmask='255.255.255.0'  />"+

                    "<interface hosts='R7' towards='N6 R6' address='192.168.6.1' netmask='255.255.255.0'  />"+
                    "<interface hosts='R4' towards='N6 R6' address='192.168.6.2' netmask='255.255.255.0'  />"+
                    "<interface hosts='R6' towards='N6 R7' address='192.168.6.3' netmask='255.255.255.0'  />"+
```

Рисунок 13 – IPv4NetworkConfigurator

Если в сети используются маршрутизаторы, то следует описать таблицу маршрутизации, либо настроить протокол маршрутизации (такой, как OSPF). Настройка маршрутизации производится в файле ASConfig.xml (рисунок 14).

```
<!-- Routers -->
<Router name="R1" RFC1583Compatible="true">
  <BroadcastInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="1" routerPriority="1" />
  <BroadcastInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="1" routerPriority="1" />
  <BroadcastInterface ifName="eth2" areaID="0.0.0.0" interfaceOutputCost="1" routerPriority="1" />
</Router>

<Router name="R2" RFC1583Compatible="true">
  <BroadcastInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="2" routerPriority="1" />
  <BroadcastInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="2" routerPriority="2" />
</Router>
```

Рисунок 14 – Пример настройки маршрутизации

Для проведения экспериментов необходимо настроить работу приложений прикладного уровня. В среде OmNET в качестве приложений-генераторов трафика используются компоненты tcpApp, udpApp, pingApp,

sctpApp. Параметры данные приложений настраиваются в файле инициализации (omnet.ini). Для приложения, работающего по транспортному протоколу TCP на клиенте можно задать время начала передачи данных, размер передаваемого файла, по какому порту соединяться с серверов, имя сервера, время конца передачи данных, задержка между соединением и началом передачи данных. Для TCP-сервера можно установить параметр принятия TCP-соединения на заданном порту (рисунок 15). Также для TCP-приложения можно определить по какому алгоритму TCP работать: TCPReno, defaultTCP, TCPVegas и т.д. (рисунок 16).

```
## tcp apps
**.numTcpApps = 1
**.client*.tcpApp[*].typename = "TCPSessionApp"
**.client*.tcpApp[0].active = true
**.client*.tcpApp[0].connectAddress = "server"
**.client*.tcpApp[0].connectPort = 1000
**.client*.tcpApp[0].tOpen = 0s
**.client*.tcpApp[0].tSend = 0s
**.client*.tcpApp[0].sendBytes = 20MB
**.client*.tcpApp[0].tClose = 0s

**.server*.tcpApp[*].typename = "TCPEchoApp"
**.server*.tcpApp[0].localPort = 1000
```

Рисунок 15 - параметры TCP-приложения

```
[Config inet-reno]
description = "TCP <---> TCP with reno algorithm"
*.server*.tcpType = "TCP"
*.client*.tcpType = "TCP"
**.tcp.tcpAlgorithmClass = "TCPReno"
*.per = 0.01 * ${0, 0.1, 0.2, 0.5, 1, 2, 5}
*.server*.tcpApp[0].echoFactor = 0


[Config inet-vegas]
description = "TCP <---> TCP with Vegas algorithm"
*.server*.tcpType = "TCP"
*.client*.tcpType = "TCP"
**.tcp.tcpAlgorithmClass = "TCPVegas"
*.per = 0.01 * ${0, 0.1, 0.2, 0.5, 1, 2, 5}
*.server*.tcpApp[0].echoFactor = 0
```

Рисунок 16 - конфигурации TCP-приложения

Для моделирования работы приложений, использующих транспортный протокол UDP, применяется компонент udpApp. Для передачи данных с помощью udpApp на клиенте настраиваются параметры: размер пакета, интервал между отправкой пакетов, получатель, время начала и конца работы приложения. Для сервера включается режим «слушать» (рисунок 17) с помощью параметра typename.

```
# udp app
**.numUdpApps = 1
**.client1.udpApp[0].typename = "UDPBasicApp"
**.client1.udpApp[0].destPort = 1234
**.client1.udpApp[0].messageLength = 1024 bytes
**.client1.udpApp[0].sendInterval = 0.00009s
**.client1.udpApp[0].startTime = 100s
**.client1.udpApp[0].stopTime = this.startTime + 10s
**.client1.udpApp[0].destAddresses = "server"
**.server.udpApp[0].typename = "UDPEchoApp"
**.server.udpApp[0].localPort = 1234
```

Рисунок 17 – UdpApp

Результатом первого этапа является модель сети, готовая к проведению экспериментов.

2.2 Этап 2 - Проведение экспериментов

Перед началом проведения эксперимента рекомендуется ограничить его длительность для избегания зацикливания. Для этого в файле omnetpp.ini требуется задать параметр sim-time-limits=<время>.

Для проведения серии однотипных экспериментов с различающимися параметрами рекомендуется создать различные конфигурации (пример см. на рисунке 16). При запуске эксперимента среда OmNET предложит выбор конфигурации.

Для записи показателей производительности сети необходимо определить точку съема сетевого трафика и настроить запись трафика. В файл omnetpp.ini нужно добавить инструкции:

<название узла>.eth[<номер интерфейса>].numOutputHooks = 1

<название узла>.eth[<номер интерфейса>].outputHook[0].typename = "ThruputMeter"

Эксперимент начинается при нажатии кнопки «запуск» в меню среды моделирования.

2.3 Этап 3 - Обработка результатов экспериментов

В результате эксперимента среда Omnet создает набор .anf файлов, содержащих экспериментальных данные. Среда Omnet содержит встроенные средства обработки .anf файлов и построения диаграмм и графиков. При запуске обработчика .anf-файлов на вкладке «BrowseData» имеется возможность просмотра записанных данных.

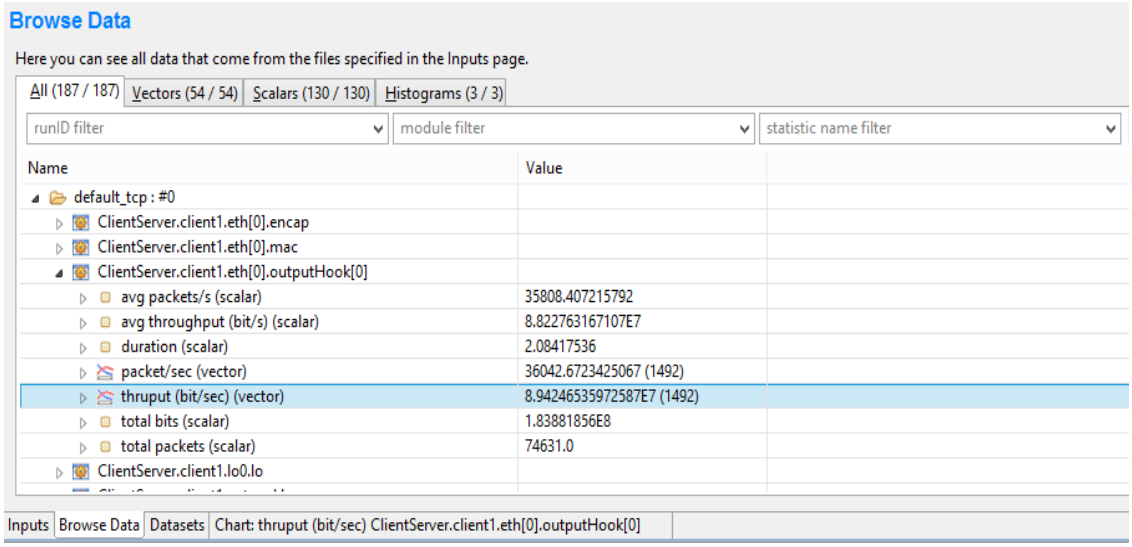

Рисунок 18 - массивы экспериментальных данных

При выборе нужного массива данных среда автоматически представляет его в виде графика.

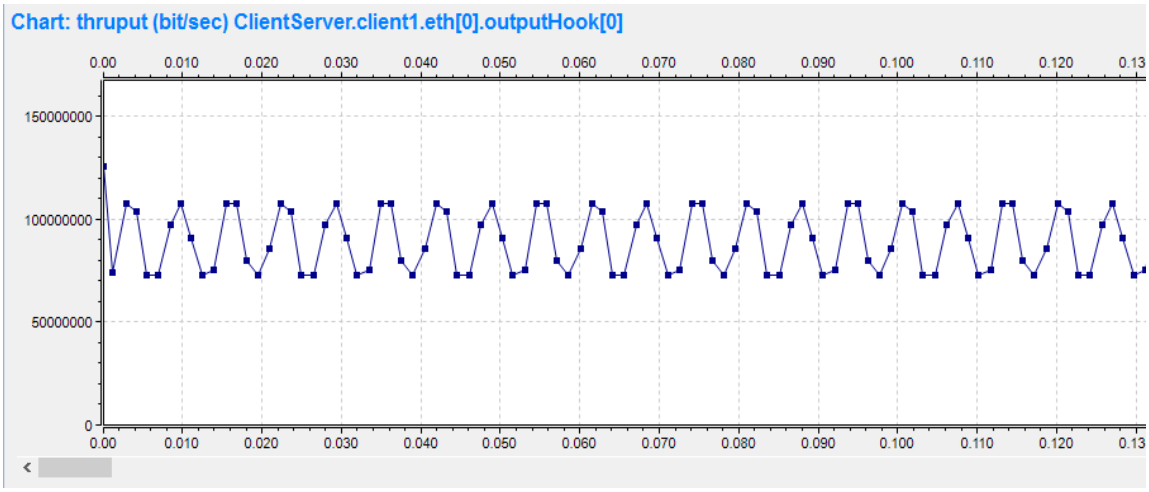

Рисунок 19 – графическое представление экспериментальных данных

2.4 Выводы по разделу

Предложенная методика содержит три основных этапа и позволяет создавать модель сети произвольной топологии, настраивать параметры приложений, профиль создаваемого трафика и другие параметры модели. Созданная модель сети полностью готова к проведению экспериментов и дальнейшей обработке экспериментальных данных. В результате экспериментов возможно достижение цели исследования – измерение производительности сети.

3 Реализация сетевых топологий и измерение производительности

3.1 Прямое соединение хостов

Ихначально разработанная методика протестирована на топологии «прямое соединение хостов». Модель сети с данной топологией включает в себя 2 модуля StandardHost, соединенные каналом связи, в качестве которого выступает модуль Eth100M с параметрами по умолчанию (datarate=100000000, delay=0, per=0, ber=0). Данная модель сети характерна тем, что результаты экспериментов можно предсказать заранее и оценить адекватность полученных результатов.

Для реализации данной модели сети был разработан набор исходных кодов, включающий в себя файлы omnetpp.ini, clientserver.ned (см. Приложения А, Б). Модель сети включает в себя несколько различных конфигураций экспериментов:

- tcpApp, алгоритм по умолчанию (TCPReno)
- tcpApp, алгоритм TCPVegas
- tcpApp, алгоритм TCPTahoe
- udpApp, размер датаграммы 1024 байта, интервалы между датаграммами 0,89 мс, 0,9 мс.

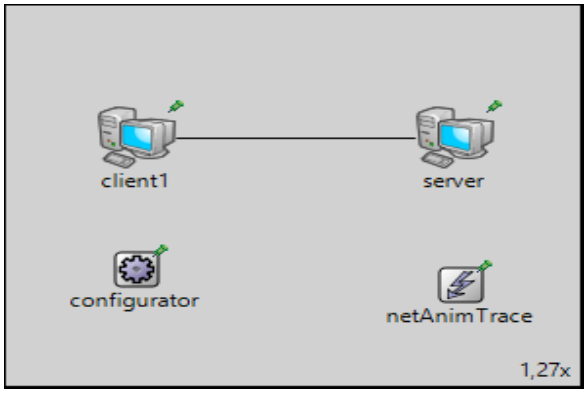

Рисунок 20 – топология сети

В последней конфигурации размеры датаграммы и интервал выбраны таким образом, чтобы суммарная скорость передаваемых через сетевой интерфейс данных (межкадровые интервалы, преамбула, Ethernet, TCP, UDP-заголовки) была более 100 МБит/с в первом случае и менее 100 МБит/с - во втором.

Для записи параметров сетевого трафика настроено две точки съема трафика — на сетевых интерфейсах обоих хостов. Трафик снимался модулем outputHook, который записывает следующие параметры трафика (см. рис. 18): средняя пакетная скорость (пакет/с), средняя скорость передачи данных (бит/с), производительность сети (скорость передачи данных прикладного уровня, бит/с). Максимальное время эксперимента ограничено значением 100с.

В результате эксперимента получена серия экспериментальных данных. Далее приведены примеры экспериментальных данных, полученных в эксперименте со второй конфигурацией (алгоритм TCPVegas).

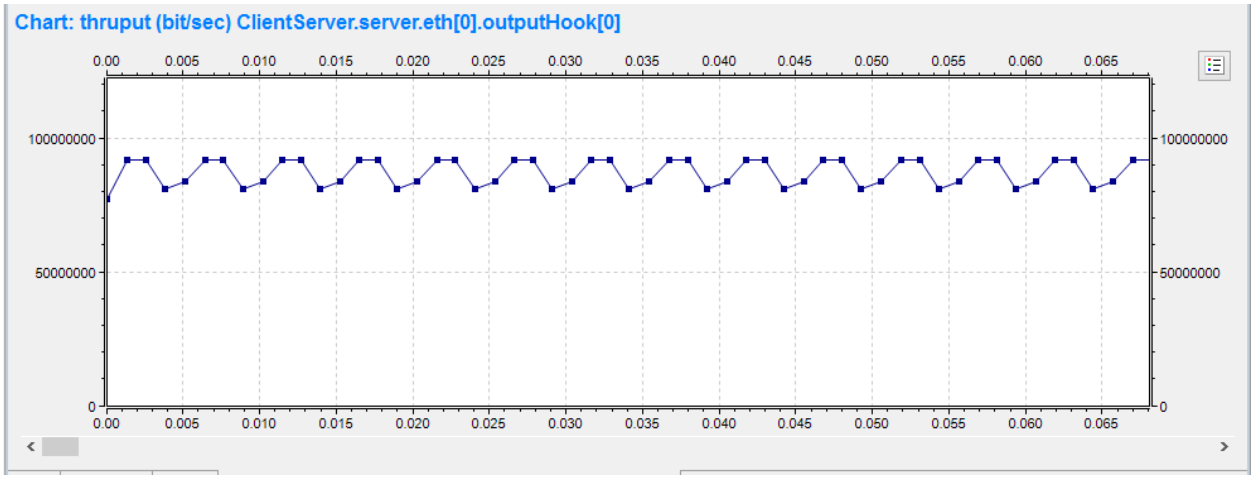

Рисунок 21 — Скорость передачи данных прикладного уровня, измеренная на отправителе трафика

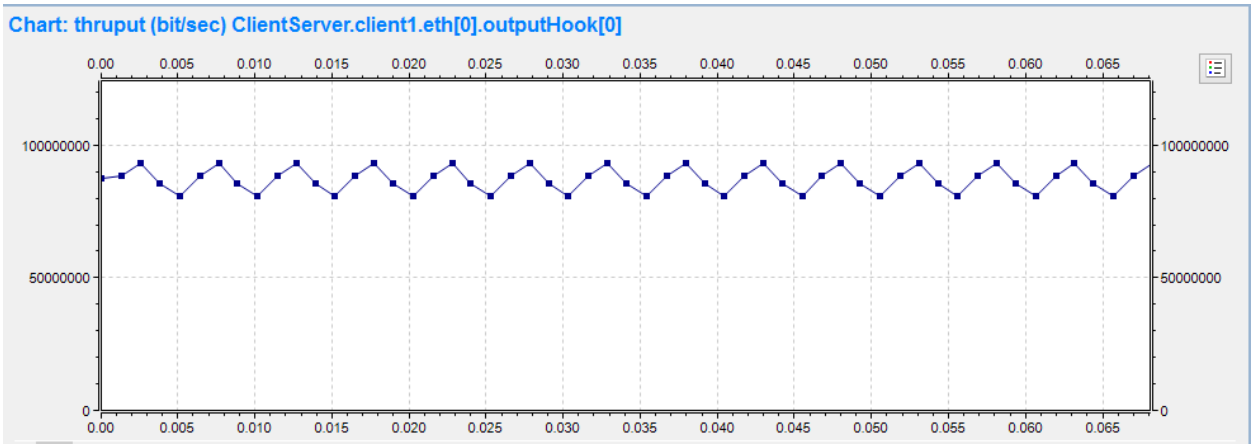

Рисунок 22 — Скорость передачи данных прикладного уровня, измеренная на получателе трафика

Таблица 2 — Средняя скорость передачи данных прикладного уровня

Точка съема трафика	Скорость передачи данных, МБит/с
client1.eth[0]	92,8023
server1.eth[0]	92,7208

В таблице 3 приведены результаты экспериментов в последней конфигурации:

Таблица 3 — Средняя скорость передачи данных прикладного уровня

Точка съема трафика	Интервал между датаграммами, мс	Скорость передачи данных, Мбит/с	Количество потерянных пакетов
client1.eth[0]	0,9	93,5104	0
server1.eth[0]	0,9	93,5104	0
client1.eth[0]	0,89	94,5618	2
server1.eth[0]	0,89	94,5610	0

3.2 Точка-точка с тремя маршрутизаторами

Далее методика протестирована на топологии «точка-точка с тремя маршрутизаторами». Модель сети с данной топологией включает в себя 2 модуля StandardHost, 3 модуля маршрутизаторов, работающих по протоколу OSPF соединенные однотипными каналами связи, в качестве которого выступает модуль Eth100M с параметрами по умолчанию (datarate=100000000, delay=0, per=0, ber=0).

Для реализации данной модели сети был разработан набор исходных кодов, включающий в себя файлы omnetpp.ini, Test.ned, ASConfig.xml (см. Приложениях В, Г, Д). Модель сети также, как и в модели «точка-точка», включает в себя несколько различных конфигураций экспериментов:

- tcpApp, алгоритм по умолчанию (TCPReno)
- tcpApp, алгоритм TCPVegas
- tcpApp, алгоритм TCPTahoe
- udpApp, размер датаграммы 1024 байта, интервалы между датаграммами 0,89 мс, 0,9 мс.

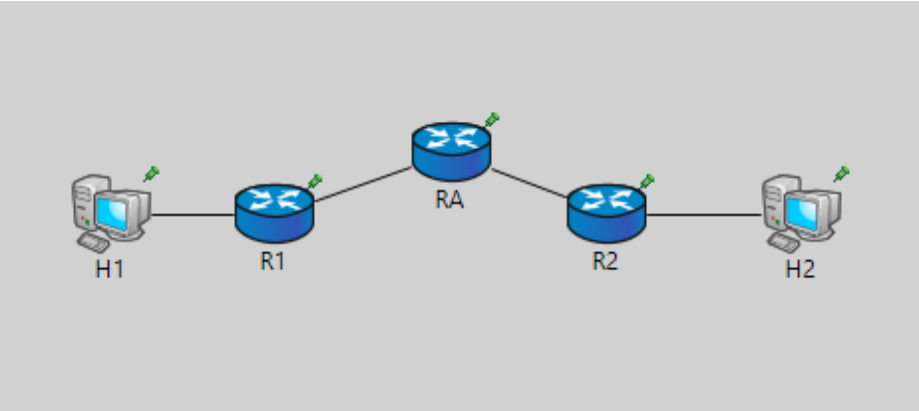

Рисунок 23 – топология сети

Точки съема трафика также находятся на сетевых интерфейсах обоих хостов. Максимальное время эксперимента ограничено значением 100с.

В результате эксперимента получена серия экспериментальных данных. Ниже приведены примеры экспериментальных данных, полученных в эксперименте со второй конфигурацией (алгоритм TCPVegas).

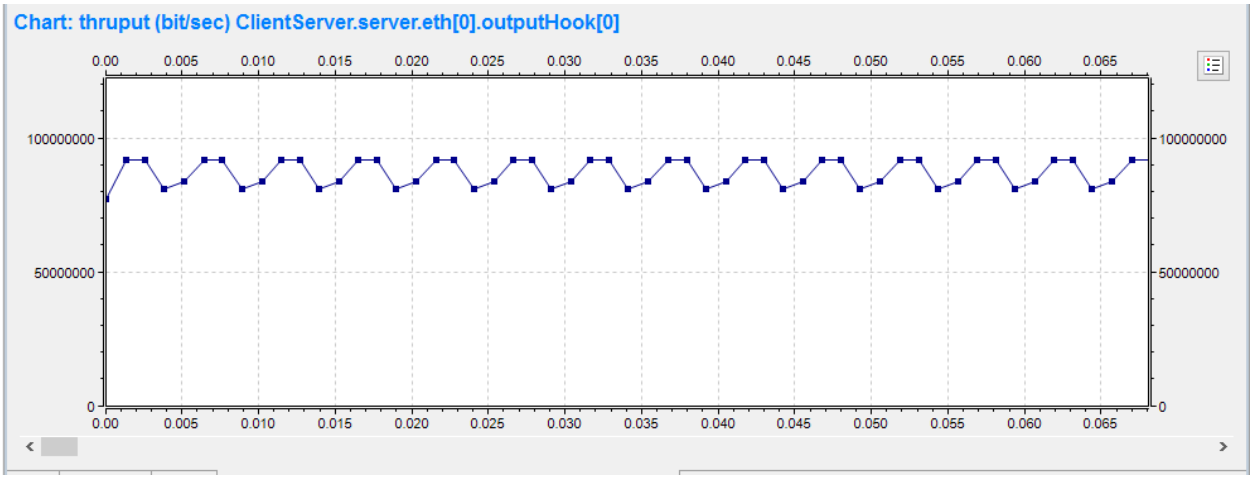

Рисунок 24 — Скорость передачи данных прикладного уровня, измеренная на отправителе трафика

Передача данных начинается не с 0 секунды, так как маршрутизаторам необходимо время для настройки, поэтому передача начинается с 9 секунды.

Таблица 4 — Средняя скорость передачи данных прикладного уровня

Точка съема трафика	Скорость передачи данных, Мбит/с
H1.eth[0]	92,7691
H2.eth[0]	92,3529

В таблице 5 приведены результаты экспериментов в последней конфигурации:

Таблица 5 — Средняя скорость передачи данных прикладного уровня

Точка съема трафика	Интервал между датаграммами, мс	Скорость передачи данных, МБит/с	Количество потерянных пакетов
H1.eth[0]	0,9	93,5103	0
H2.eth[0]	0,9	93,5103	0
H1.eth[0]	0,89	94,5618	8
H2.eth[0]	0,89	94,4748 МБит/с	0

3.3 Дерево с 6 маршрутизаторами и конкуренцией за канал

Для тестирования методики топологией следующей модели применялась конфигурация «дерево с 6 маршрутизаторами и конкуренцией за канал». Модель сети с данной топологией включает в себя 4 модуля StandardHost, соединенные однотипными каналами связи, в качестве которого выступает модуль Eth100M с параметрами по умолчанию, 6 маршрутизаторов работающих по протоколу OSPF. Хосты H1 и H2 обмениваются контрольным трафиком, в то время как H3 и H4 создают нагрузочный трафик.

Для реализации данной модели сети был разработан набор исходных кодов, включающий в себя файлы omnetpp.ini, Test.ned, ASConfig.xml (см. Приложение Е, Ж, И). Модель сети включает в себя несколько различных конфигураций экспериментов:

- tcpApp, алгоритм по умолчанию (TCPReno)
- tcpApp, алгоритм TCPVegas
- tcpApp, алгоритм TCPTahoe
- udpApp, размер датаграммы 1024 байта, интервалы между датаграммами 0,89 мс, 0,9 мс.

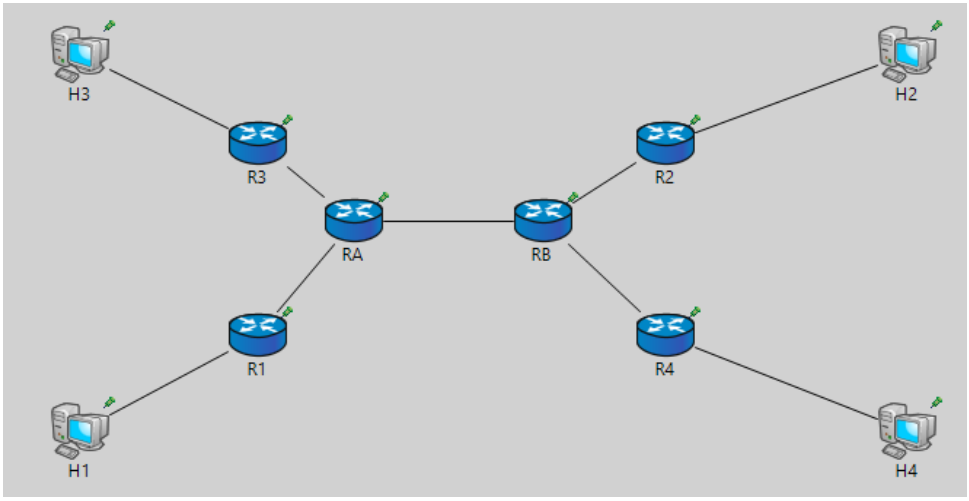

Рисунок 25 – топология сети

Для записи параметров сетевого трафика настроено две точки съема трафика — на сетевых интерфейсах хостов H1 и H2. Максимальное время эксперимента ограничено значением 100с. В результате эксперимента получена серия экспериментальных данных. Ниже приведены примеры экспериментальных данных по второй конфигурации.

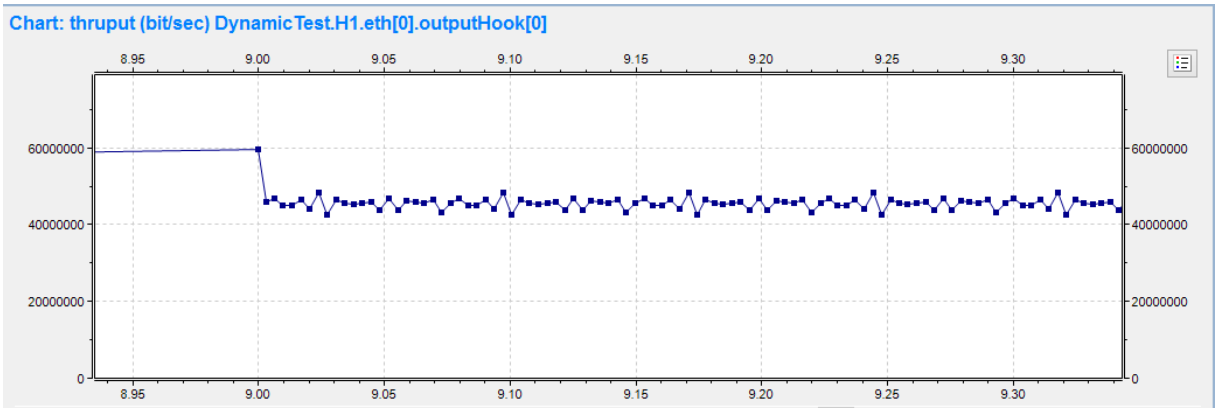

Рисунок 26 — Скорость передачи данных прикладного уровня, измеренная на отправителе трафика

В таблице 6 приведены средние значения скорости для приложения TCP.

Таблица 6 — Средняя скорость передачи данных прикладного уровня

Точка съема трафика	Скорость передачи данных, МБит/с
H1.eth[0]	53,3271
H2.eth[0]	53,2308

В таблице 7 приведены результаты экспериментов в четвертой конфигурации.

Таблица 7 — Средняя скорость передачи данных прикладного уровня

Точка съема трафика	Интервал между датаграммами, мс	Скорость передачи данных, МБит/с	Количество потерянных пакетов
client1.eth[0]	0,9	86,4002	0
server1.eth[0]	0,9	86,3717	0
client1.eth[0]	0,89	94,0015	8
server1.eth[0]	0,89	94,0010	0

3.4 Топология, приближенная к реальным сетям

Четвертая топология может использоваться в реальных сетях. Модель сети с данной топологией включает в себя 10 модулей StandardHost, соединенные однотипным каналом связи, в качестве которого выступает модуль Eth100M с параметрами по умолчанию, 6 коммутаторов EtherSwitch, 6 маршрутизаторов работающих по протоколу OSPF.

Для реализации данной модели сети был разработан набор исходных кодов, включающий в себя файлы omnetpp.ini, Test.ned, ASConfig.xml (см. Приложение К, Л, М). Модель сети включает в себя несколько различных конфигураций экспериментов:

- tcpApp, алгоритм по умолчанию (TCPReno)

- udpApp, размер датаграммы 1024 байта, интервалы между датаграммами 0,89 мс, 0,9 мс.

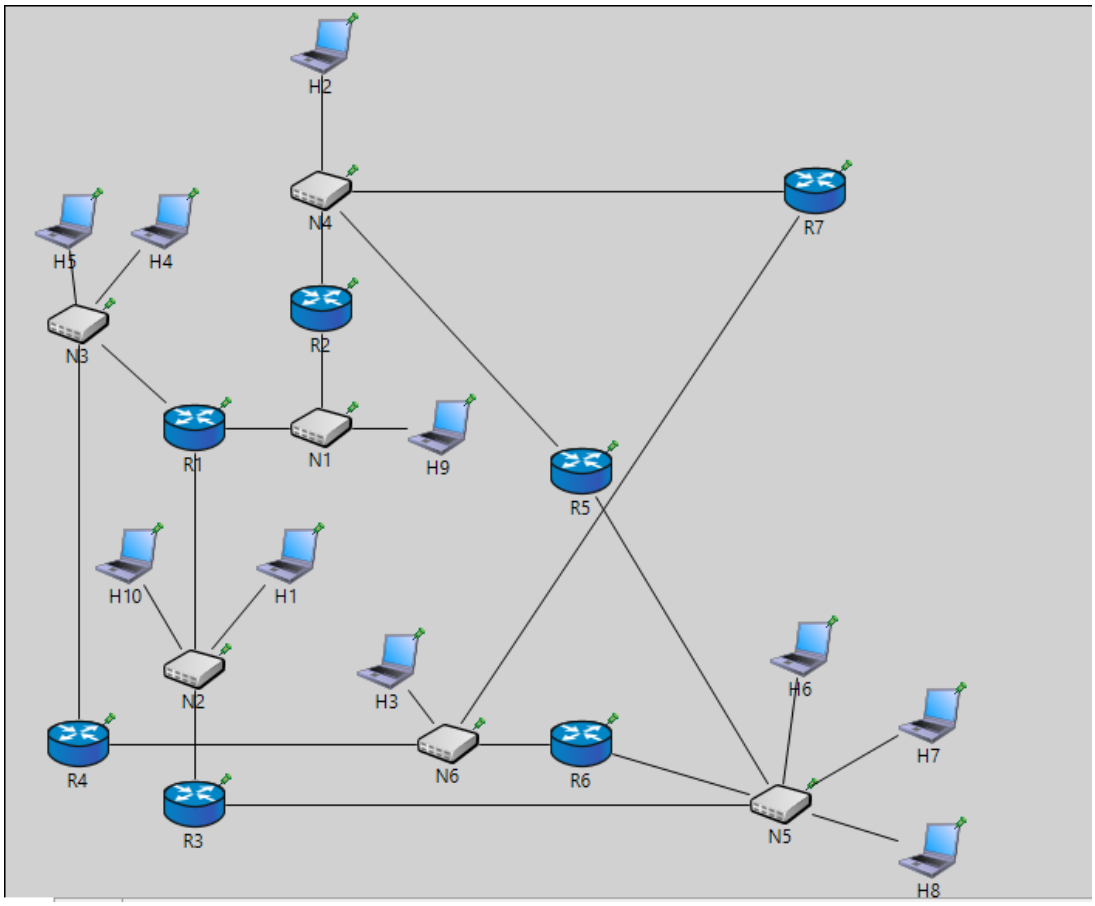

Рисунок 26 – топология сети

В последней конфигурации размеры датаграммы и интервал выбраны таким образом, чтобы суммарная скорость передаваемых через сетевой интерфейс данных (межкадровые интервалы, преамбула, Ethernet, TCP, UDP-заголовки) была более 100 МБит/с в первом случае и менее 100 МБит/с - во втором.

Для записи параметров сетевого трафика настроены точки съема трафика на сетевых интерфейсах хостов H1 и H2. Трафик снимался модулем outputHook. Максимальное время эксперимента ограничено значением 100с.

В результате эксперимента получена серия экспериментальных данных (см. Приложение К, Л, М).

Таблица 8 — Средняя скорость передачи данных прикладного уровня

Точка съема трафика	Скорость передачи данных, МБит/с
H1.eth[0]	89,7112
H2.eth[0]	89,7008

В таблице 9 приведены результаты экспериментов в четвертой конфигурации:

Таблица 9 — Средняя скорость передачи данных прикладного уровня

Точка съема трафика	Интервал между датаграммами, мс	Скорость передачи данных, МБит/с	Количество потерянных пакетов
H1.eth[0]	0,9	93,5103	353
H2.eth[0]	0,9	93,5102	0
H1.eth[0]	0,89	94,5618	629
H2.eth[0]	0,89	94,5611	0
H1.eth[0]	1	92,4756	153
H2.eth[0]	1	92,4687	0
H1.eth[0]	1,1	91,1046	22
H2.eth[0]	1,1	90,9887	0
H1.eth[0]	1,2	90,0112	1
H2.eth[0]	1,2	89,9584	0

4. Анализ полученных результатов

На данном этапе выполнена оценка достоверности результатов экспериментов, полученных по разработанной методике. Эксперимент, описанный в главе 3.1 поставлен таким образом, что его результаты легко предсказать теоретически и сравнить с полученными экспериментальными данными.

В эксперименте в конфигурации 4 (udpApp) исследовалась максимально достижимая скорость передачи данных по каналу Ethernet 100 Мбит/с. За исходные данные возьмем размер UDP-датаграммы в 1024 байт, размеры служебных заголовков составляют:

- 8 байт - UDP-заголовок
- 20 байт - IP-заголовок
- 14 байт - Ethernet-заголовок, 4 байта - контрольная сумма
- 8 байт - Ethernet-преамбула
- 96 бит (12 байт) - межкадровый интервал

Таким образом, полный размер кадра составляет 1086 байт. Отсюда, максимально достижимая скорость передачи полезных данных составляет:

100Мбит/с * 1024 / 1086 = 94,29 Мбит/с

Данное значение хорошо согласуется с результатом эксперимента (94,56 МБит/с). Ошибка составила 0,2%, что является хорошим показателем. Отсюда следует, что разработанная методика позволяет измерять производительность сети с высокой точностью.

В эксперименте в конфигурациях 1-3 (tcpApp с различными алгоритмами) получены следующие показатели производительности:

- 92,76Мбит/с при использовании алгоритма TCPReno
- 91,11 Мбит/с при использовании алгоритма TCPVegas
- 92,25 Мбит/с при использовании алгоритма TCPTahoe

Теоретический расчет производительности сети при использовании TCP-приложений является сложной задачей. Сравним полученные показатели с результатами других экспериментов:

- 94,1 Мбит/с в эксперименте с использованием iperf [9]
- 94 Мбит/с в натурных экспериментах [10]

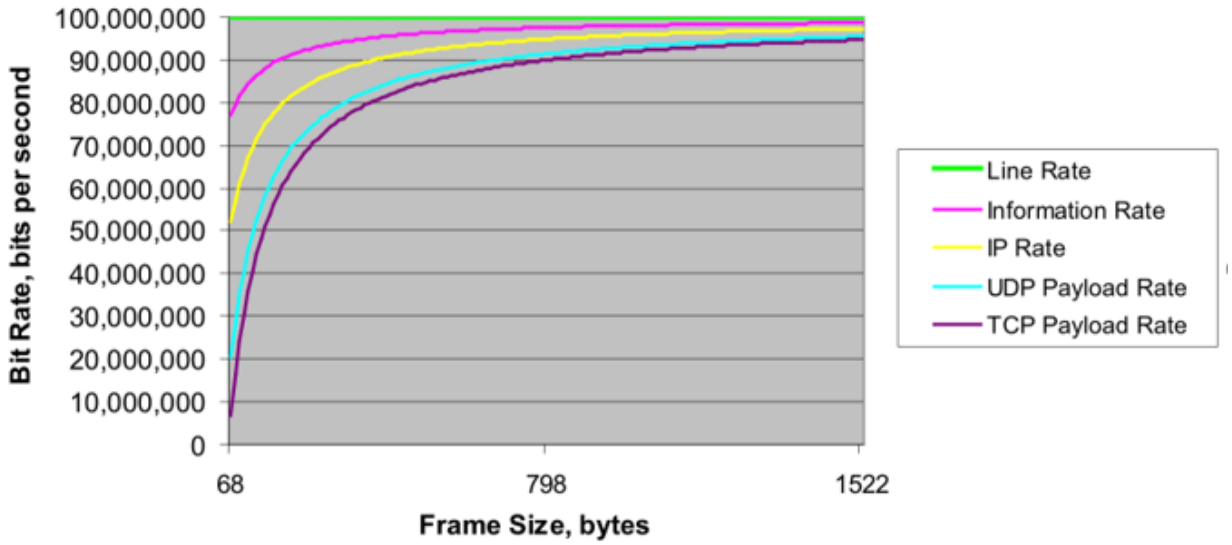

Рисунок 27 – результаты натурных экспериментов

4.1 Выводы по разделу

Показатели производительности, полученные в результате эксперимента в среде Omnet++, хорошо согласуются с результатами других экспериментов. Относительная ошибка составляет 1-2%.

Отсюда можно сделать вывод, что показатели производительности, полученные в ходе экспериментов над моделью сети в среде Omnet++ по разработанной методике достаточно точно предсказывают производительность реальной сети.

ЗАКЛЮЧЕНИЕ

В работе рассмотрены методики измерения производительности сети, реализованные в автоматизированных инструментах, таких, как iperf. Проведен обзор среды моделирования Omnet++, выбраны компоненты, с помощью которых можно реализовать измерение производительности. Разработана методика измерения производительности сети с использованием Omnet++. Созданы модели сетей с различными топологиями, разработанная методика протестирована на данных моделях. Собран достаточный для подтверждения объем экспериментальных данных. Проведенный анализ экспериментальных показывает, что показатели производительности, полученные по разработанной методике, достаточно точно предсказывают производительность реальных сетей.

Список использованных источников

1. Олифер, В. Г. Компьютерные сети. Принципы, технологии, протоколы. / В. Г. Олифер, Н. А. Олифер – Санкт-Петербург: Питер, 2010. – 944 с.
2. Определение локальных сетей и их топология; основы локальных сетей - НИЯУ «МИФИ» [Электронный ресурс] – Режим доступа: http://www.intuit.ru/studies/courses/57/57/lecture/1672?page=1
3. Что такое FTP [Электронный ресурс] – Режим доступа: https://semantica.in/blog/ftp.html
4. Таненбаум, Э. Компьютерные сети 5-е издание./ Таненбаум Э, Уэзеролл Д – Санкт-Петербург: Питер, 2012. – 960 с.
5. Комер, Д. Межсетевой обмен с помощью TCPIP [Электронный ресурс] – Режим доступа: http://www.citforum.ru/internet/comer/contents.shtml
6. Виснадул, Б. Д. Основы компьютерных сетей. / Б. Д. Виснадул, С. А. Лупин, С. В. Сидоров, П. Ю. Чумаченко. — Москва: Форум, Инфра-М, 2000. – 272 с.
7. Хогдал, Дж. - Анализ и диагностика компьютерных сетей. / Дж. С. Хогдал – Москва: Лори, 2001. – 155 с.
8. Назаров, А. Н. Технические решения создания сетей. Справочное издание. / А. Н. Назаров, И. А. Разживин, М. В. Симонов. – Москва: Горячая линия – Телеком, 2013. – 376 с.
9. Network Performance Measurements with iperf [Электронный ресурс] – Режим доступа: https://sandilands.info/sgordon/teaching/resources/iperf.html
10. Understanding Carrier Ethernet Throughput [Электронный ресурс] – Режим доступа: https://www.mef.net/Assets/White_Papers/Understanding_Carrier_Ethernet_Throughput_-_v14.pdf

11. OMNeT++ [Электронный ресурс] : OMNeT++ Discrete Event Simulator. – Режим доступа: https://omnetpp.org

12. Олифер Н. А. Средства анализа и оптимизации локальных сетей [Электронный ресурс] / Н. А. Олифер, В. Г. Олифер // Центр Информационных Технологий. – 1998. – Режим доступа: http://citforum.ru/nets/optimize/index.shtml

13. INET Framework User's Guide [Электронный ресурс] : INET Framework. – Режим доступа: https://inet.omnetpp.org/users-guide/inet-users-guide.pdf

14. LAN Speed Test Guide [Электронный ресурс] : Totusoft Lan Speed test – Режим доступа: https://totusoft.com/lanspeed

15. Lanbench tcp network benchmark [Электронный ресурс] –Режим доступа: http://www.zachsaw.com/?pg=lanbench_tcp_network_benchmark

16. Iperf users docs [Электронный ресурс] – Режим доступа: https://iperf.fr/iperf-doc.php

ПРИЛОЖЕНИЕ А

Содержимое файла clientserver.ned, описывающего топологию модели тестовой сети «точка-точка»

```
package inet.examples.inet.tcpclientserver;
import inet.networklayer.configurator.ipv4.IPv4NetworkConfigurator;
import inet.node.ethernet.Eth100M;
import inet.node.inet.StandardHost;
network ClientServer
{
    parameters:
        double per = default(0);
        @display("bgb=232,193");
    types:
    submodules:
        client1: StandardHost {
            parameters:
                @display("p=53,67");
        }
        server: StandardHost {
            parameters:
                @display("p=181,67;i=device/pc2");
        }
        configurator: IPv4NetworkConfigurator {
            parameters:
                @display("p=53,134");
        }
    connections:
        client1.ethg++ <--> Eth100M <--> server.ethg++;
}
```

ПРИЛОЖЕНИЕ Б

Содержимое файла omnetpp.ini, описывающего функционирование модели тестовой сети «точка-точка»

```
[Config default_tcp]
description = "default_TCP <---> default_TCP"
**.numTcpApps = 1
**.client*.tcpApp[0].typename = "TCPSessionApp"
**.client*.tcpApp[0].active = true
**.client*.tcpApp[0].connectAddress = "server"
**.client*.tcpApp[0].connectPort = 1000
**.client*.tcpApp[0].tOpen = 0s
**.client*.tcpApp[0].tSend = 0s
**.client*.tcpApp[0].sendBytes = 20MB
**.client*.tcpApp[0].tClose = 0s
**.server*.tcpApp[0].typename = "TCPEchoApp"
**.server*.tcpApp[0].localPort = 1000
[Config inet-vegas]
description = "TCP <---> TCP with Vegas algorithm"
*.server*.tcpType = "TCP"
*.client*.tcpType = "TCP"
**.tcp.tcpAlgorithmClass = "TCPVegas"
*.server*.tcpApp[0].echoFactor = 1
**.numTcpApps = 1
**.client*.tcpApp[*].typename = "TCPSessionApp"
**.client*.tcpApp[0].active = true
**.client*.tcpApp[0].connectAddress = "server"
**.client*.tcpApp[0].connectPort = 1000
**.client*.tcpApp[0].tOpen = 0s
**.client*.tcpApp[0].tSend = 0s
**.client*.tcpApp[0].sendBytes = 120MB
```

```
**.client*.tcpApp[0].tClose = 0s
**.server*.tcpApp[*].typename = "TCPEchoApp"
**.server*.tcpApp[0].localPort = 1000
[Config inet-tahoe]
description = "TCP <---> TCP with Tahoe algorithm"
*.server*.tcpType = "TCP"
*.client*.tcpType = "TCP"
**.tcp.tcpAlgorithmClass = "TCPTahoe"
**.numTcpApps = 1
**.client*.tcpApp[*].typename = "TCPSessionApp"
**.client*.tcpApp[0].active = true
**.client*.tcpApp[0].connectAddress = "server"
**.client*.tcpApp[0].connectPort = 1000
**.client*.tcpApp[0].tOpen = 0s
**.client*.tcpApp[0].tSend = 0s
**.client*.tcpApp[0].sendBytes = 20MB
**.client*.tcpApp[0].tClose = 0s
**.server*.tcpApp[*].typename = "TCPEchoApp"
**.server*.tcpApp[0].localPort = 1000
[Config udp]
description = "With UDP"
**.numUdpApps = 1
**.client1.udpApp[0].typename = "UDPBasicApp"
**.client1.udpApp[0].destPort = 1234
**.client1.udpApp[0].messageLength = 1024 bytes
**.client1.udpApp[0].sendInterval = 0.000083s
**.client1.udpApp[0].startTime = 0s
**.client1.udpApp[0].stopTime = this.startTime + 10s
**.client1.udpApp[0].destAddresses = "server"
**.server.udpApp[0].typename = "UDPEchoApp"
```

```
**.server.udpApp[0].localPort = 1234
[Config udp_ver]
description = "With UDP"
**.numUdpApps = 1
**.client1.udpApp[0].typename = "UDPBasicApp"
**.client1.udpApp[0].destPort = 1234
**.client1.udpApp[0].messageLength = 1024 bytes
**.client1.udpApp[0].sendInterval = 0.000089s
**.client1.udpApp[0].startTime = 0s
**.client1.udpApp[0].stopTime = this.startTime + 10s
**.client1.udpApp[0].destAddresses = "server"
**.server.udpApp[0].typename = "UDPEchoApp"
**.server.udpApp[0].localPort = 1234
[General]
network = ClientServer
total-stack = 7MiB
tkenv-plugin-path = ../../../etc/plugins
sim-time-limit = 100s
**.eth[*].numOutputHooks = 1
**.eth[*].outputHook[0].typename = "ThruputMeter"
**.eth[*].mac.mtu = 9000B
*.configurator.config=xml("<config><interface                    hosts='*'
address='192.168.1.x' netmask='255.255.255.0'/></config>")
**.tun[*].mtu = 9000B
```

ПРИЛОЖЕНИЕ В

Содержимое файла Test.ned, описывающего топологию модели тестовой сети «точка-точка с тремя маршрутизаторами»

```
package inet.test;
import inet.common.lifecycle.LifecycleController;
import inet.common.misc.ThruputMeteringChannel;
import inet.networklayer.configurator.ipv4.IPv4NetworkConfigurator;
import inet.node.inet.StandardHost;
import inet.node.ospfv2.OSPFRouter;
import inet.transportlayer.udp.UDP;
network Test
{
    parameters:
        @display("p=10,10;b=712,152;bgb=720,448");
    types:
        channel C extends ThruputMeteringChannel
        {
            datarate = 100Mbps;
            thruputDisplayFormat = "#N";
        }
    submodules:
        H1: StandardHost {
            parameters:
                @display("p=110,123");
            gates:
                ethg[1];
        }
        R1: OSPFRouter {
            parameters:
```

```
            @display("p=192,123");
        gates:
            ethg[2];
    }
    RA: OSPFRouter {
        parameters:
            @display("p=281,92");
        gates:
            ethg[2];
    }
    R2: OSPFRouter {
        parameters:
            @display("p=359,123");
        gates:
            ethg[2];
    }
    H2: StandardHost {
        parameters:
            @display("p=457,123");
        gates:
            ethg[1];
    }
    configurator: IPv4NetworkConfigurator {
        parameters:
            @display("p=281,302");
            config = xml("<config>"
+ "<interface among='H1 R1' address='192.168.1.x' netmask='255.255.255.x' />"
+ "<interface among='H2 R2' address='192.168.2.x' netmask='255.255.255.x' />"
        + "<interface among='R*' address='10.0.0.x' netmask='255.255.255.x' />"
        + "<multicast-group hosts='R*' address='224.0.0.5 224.0.0.6' />"
```

```
+ "<route hosts='H1' destination='*' gateway='R1'/>"
+ "<route hosts='H2' destination='*' gateway='R2'/>"
+ "</config>");
        }
        lifecycleController: LifecycleController {
            @display("p=94,302");
        }
    connections:
        H1.ethg[0] <--> Eth100M <--> R1.ethg[0];
        R1.ethg[1] <--> Eth100M <--> RA.ethg[0];
        RA.ethg[1] <--> Eth100M <--> R2.ethg[1];
        R2.ethg[0] <--> Eth100M <--> H2.ethg[0];
}
```

ПРИЛОЖЕНИЕ Г

Содержимое файла omnetpp.ini, описывающего функционирование модели тестовой сети «точка-точка с тремя маршрутизаторами»

```
[Config default_tcp]
description = "default_TCP <---> default_TCP"
**.numTcpApps = 1
**.H1.tcpApp[0].typename = "TCPSessionApp"
**.H1.tcpApp[0].active = true
**.H1.tcpApp[0].connectAddress = "H2"
**.H1.tcpApp[0].connectPort = 1000
**.H1.tcpApp[0].tOpen = 0s
**.H1.tcpApp[0].tSend = 0.01s
**.H1.tcpApp[0].sendBytes = 20MB
**.H1.tcpApp[0].tClose = 0s
**.H2.tcpApp[0].typename = "TCPEchoApp"
**.H2.tcpApp[0].localPort = 1000
[Config inet-vegas]
description = "TCP <---> TCP with Vegas algorithm"
**.tcp.tcpAlgorithmClass = "TCPVegas"
**.numTcpApps = 1
**.H1.tcpApp[*].typename = "TCPSessionApp"
**.H1.tcpApp[0].active = true
**.H1.tcpApp[0].connectAddress = "H2"
**.H1.tcpApp[0].connectPort = 1000
**.H1.tcpApp[0].tOpen = 0s
**.H1.tcpApp[0].tSend = 0s
**.H1.tcpApp[0].sendBytes = 120MB
**.H1.tcpApp[0].tClose = 0s
**.H2.tcpApp[*].typename = "TCPEchoApp"
```

**.H2.tcpApp[0].localPort = 1000

[Config inet-tahoe]

description = "TCP <---> TCP with Tahoe algorithm"

*.H2.tcpType = "TCP"

*.H1.tcpType = "TCP"

**.tcp.tcpAlgorithmClass = "TCPTahoe"

**.numTcpApps = 1

**.H1.tcpApp[*].typename = "TCPSessionApp"

**.H1.tcpApp[0].active = true

**.H1.tcpApp[0].connectAddress = "H2"

**.H1.tcpApp[0].connectPort = 1000

**.H1.tcpApp[0].tOpen = 0s

**.H1.tcpApp[0].tSend = 0s

**.H1.tcpApp[0].sendBytes = 20MB

**.H1.tcpApp[0].tClose = 0s

**.H2.tcpApp[*].typename = "TCPEchoApp"

**.H2.tcpApp[0].localPort = 1000

[Config udp]

description = "With UDP"

**.numUdpApps = 1

**.H1.udpApp[0].typename = "UDPBasicApp"

**.H1.udpApp[0].destPort = 1234

**.H1.udpApp[0].messageLength = 1024 bytes

**.H1.udpApp[0].sendInterval = 0.000089s

**.H1.udpApp[0].startTime = 0s

**.H1.udpApp[0].stopTime = this.startTime + 10s

**.H1.udpApp[0].destAddresses = "H2"

**.H2.udpApp[0].typename = "UDPEchoApp"

**.H2.udpApp[0].localPort = 1234

[Config udp_ver]

```
description = "With UDP"
**.numUdpApps = 1
**.H1.udpApp[0].typename = "UDPBasicApp"
**.H1.udpApp[0].destPort = 1234
**.H1.udpApp[0].messageLength = 1024 bytes
**.H1.udpApp[0].sendInterval = 0.00009s
**.H1.udpApp[0].startTime = 0s
**.H1.udpApp[0].stopTime = this.startTime + 10s
**.H1.udpApp[0].destAddresses = "H2"
**.H2.udpApp[0].typename = "UDPEchoApp"
**.H2.udpApp[0].localPort = 1234
[General]
network = Test
tkenv-plugin-path = ../../../etc/plugins
sim-time-limit = 100s
**.eth[*].numOutputHooks = 1
**.eth[*].outputHook[0].typename = "ThruputMeter"
**.ospf.ospfConfig = xmldoc("ASConfig.xml")
**.arp.cacheTimeout = 1s
```

ПРИЛОЖЕНИЕ Д

Содержимое файла ASConfig.xml, описывающего маршрутизацию в модели тестовой сети «точка-точка с тремя маршрутизаторами»

```
<?xml version="1.0"?>
<OSPFASConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="OSPF.xsd">
 <!-- Areas -->
 <Area id="0.0.0.0">
  <AddressRange address="RA>R1" mask="RA>R1" status="Advertise" />
  <AddressRange address="RA>R2" mask="RA>R2" status="Advertise" />
 </Area>
 <Area id="0.0.0.1">
  <AddressRange address="H1" mask="H1" status="Advertise" />
 </Area>
 <Area id="0.0.0.2">
  <AddressRange address="H2" mask="H2" status="Advertise" />
 </Area>
  <!-- Routers -->
 <Router name="R1" RFC1583Compatible="true">
<BroadcastInterface ifName="eth0" areaID="0.0.0.1" interfaceOutputCost="1" />
<PointToPointInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="2" />
 </Router>
 <Router name="R2" RFC1583Compatible="true">
<BroadcastInterface ifName="eth0" areaID="0.0.0.2" interfaceOutputCost="1" />
<PointToPointInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="2" />
```

```
        </Router>
        <Router name="RA" RFC1583Compatible="true">
<PointToPointInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="2"
/>
<PointToPointInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="2"
/>
        </Router>
      </OSPFASConfig>
```

ПРИЛОЖЕНИЕ Е

Содержимое файла Test.ned, описывающего топологию модели тестовой сети «дерево с 6 маршрутизаторами и борьбой за канал»

```
package inet.ospfv2.dynamictest;
import inet.common.lifecycle.LifecycleController;
import inet.common.misc.ThruputMeteringChannel;
import inet.networklayer.configurator.ipv4.IPv4NetworkConfigurator;
import inet.node.inet.StandardHost;
import inet.node.ospfv2.OSPFRouter;
network Test
{
    parameters:
        @display("p=10,10;b=712,152;bgb=720,448");
    types:
        channel C extends ThruputMeteringChannel
        {
            delay = 0.1us;
            datarate = 100Mbps;
            thruputDisplayFormat = "#N";
        }
    submodules:
        H1: StandardHost {
            parameters:
                @display("p=84,363");
            gates:
                ethg[1];
        }
        R1: OSPFRouter {
            parameters:
```

```
            @display("p=206,296");
        gates:
            ethg[2];
    }
    RA: OSPFRouter {
        parameters:
            @display("p=272,216");
        gates:
            ethg[3];
    }
    R2: OSPFRouter {
        parameters:
            @display("p=486,162");
        gates:
            ethg[2];
    }
    H2: StandardHost {
        parameters:
            @display("p=652,100");
        gates:
            ethg[1];
    }
    configurator: IPv4NetworkConfigurator {
        parameters:
            @display("p=143,19");
            config = xml("<config>"
 + "<interface among='H1 R1' address='192.168.1.x' netmask='255.255.255.x' />"
+ "<interface among='H2 R2' address='192.168.2.x' netmask='255.255.255.x' />"
+ "<interface among='H3 R3' address='192.168.3.x' netmask='255.255.255.x' />"
+ "<interface among='H4 R4' address='192.168.4.x' netmask='255.255.255.x' />"
```

```
+ "<interface among='R*' address='10.0.0.x' netmask='255.255.255.x' />"
+ "<multicast-group hosts='R*' address='224.0.0.5 224.0.0.6' />"
+ "<route hosts='H1' destination='*' gateway='R1'/>"
+ "<route hosts='H2' destination='*' gateway='R2'/>"
+ "<route hosts='H3' destination='*' gateway='R3'/>"
+ "<route hosts='H4' destination='*' gateway='R4'/>"
+ "</config>");
    }
    lifecycleController: LifecycleController {
        @display("p=53,19");
    }
    H3: StandardHost {
        parameters:
            @display("p=84,100");
        gates:
            ethg[1];
    }
    H4: StandardHost {
        parameters:
            @display("p=652,363");
        gates:
            ethg[1];
    }
    R3: OSPFRouter {
        parameters:
            @display("p=206,162");
        gates:
            ethg[2];
    }
    R4: OSPFRouter {
```

```
        parameters:
            @display("p=486,296");
        gates:
            ethg[2];
    }
    RB: OSPFRouter {
        parameters:
            @display("p=402,216");
        gates:
            ethg[3];
    }
connections: H1.ethg[0] <--> Eth100M <--> R1.ethg[0];
    R1.ethg[1] <--> Eth100M <--> RA.ethg[0];
    R2.ethg[0] <--> Eth100M <--> H2.ethg[0];
    RA.ethg[2] <--> Eth100M <--> R3.ethg[1];
    H3.ethg[0] <--> Eth100M <--> R3.ethg[0];
    H4.ethg[0] <--> Eth100M <--> R4.ethg[0];
    RA.ethg[1] <--> Eth100M <--> RB.ethg[0];
    R2.ethg[1] <--> Eth100M <--> RB.ethg[1];
    R4.ethg[1] <--> Eth100M <--> RB.ethg[2];
```

ПРИЛОЖЕНИЕ Ж

Содержимое файла omnetpp.ini, описывающего функционирование модели тестовой сети «дерево с 6 маршрутизаторами и борьбой за канал»

```
[Config default_tcp]
description = "default_TCP <---> default_TCP"
**.numTcpApps = 1
**.H1.tcpApp[0].typename = "TCPSessionApp"
**.H1.tcpApp[0].active = true
**.H1.tcpApp[0].connectAddress = "H2"
**.H1.tcpApp[0].connectPort = 1000
**.H1.tcpApp[0].tOpen = 0s
**.H1.tcpApp[0].tSend = 0s
**.H1.tcpApp[0].sendBytes = 20MB
**.H1.tcpApp[0].tClose = 0s
**.H2.tcpApp[0].typename = "TCPEchoApp"
#**.server*.tcpApp[*].echoDelay = 0.0000000001s
**.H2.tcpApp[0].localPort = 1000
[Config inet-vegas]
description = "TCP <---> TCP with Vegas algorithm"
*.H1.tcpType = "TCP"
*.H2.tcpType = "TCP"
**.tcp.tcpAlgorithmClass = "TCPVegas"
**.numTcpApps = 1
**.H1.tcpApp[*].typename = "TCPSessionApp"
**.H1.tcpApp[0].active = true
**.H1.tcpApp[0].connectAddress = "H2"
**.H1.tcpApp[0].connectPort = 1000
**.H1.tcpApp[0].tOpen = 0s
**.H1.tcpApp[0].tSend = 0.01s
```

```
**.H1.tcpApp[0].sendBytes = 120MB
**.H1.tcpApp[0].tClose = 0s
**.H2.tcpApp[*].typename = "TCPEchoApp"
**.H2.tcpApp[0].localPort = 1000
[Config inet-tahoe]
description = "TCP <---> TCP with Tahoe algorithm"
*.H2.tcpType = "TCP"
*.H1.tcpType = "TCP"
**.tcp.tcpAlgorithmClass = "TCPTahoe"
**.numTcpApps = 1
**.H1.tcpApp[*].typename = "TCPSessionApp"
**.H1.tcpApp[0].active = true
**.H1.tcpApp[0].connectAddress = "H2"
**.H1.tcpApp[0].connectPort = 1000
**.H1.tcpApp[0].tOpen = 0s
**.H1.tcpApp[0].tSend = 0.01s
**.H1.tcpApp[0].sendBytes = 20MB
**.H1.tcpApp[0].tClose = 0s
**.H2.tcpApp[*].typename = "TCPEchoApp"
**.H2.tcpApp[0].localPort = 1000
[Config udp]
description = "With UDP"
**.H1.numUdpApps = 1
**.H1.udpApp[0].typename = "UDPBasicApp"
**.H1.udpApp[0].destPort = 1234
**.H1.udpApp[0].messageLength = 512 bytes
**.H1.udpApp[0].sendInterval = 0.00005s
**.H1.udpApp[0].startTime = 0s
**.H1.udpApp[0].stopTime = this.startTime + 10s
**.H1.udpApp[0].destAddresses = "H2"
```

```
**.H2.numUdpApps = 1
**.H2.udpApp[0].typename = "UDPEchoApp"
**.H2.udpApp[0].localPort = 1234
**.H3.numUdpApps = 1
**.H3.udpApp[0].typename = "UDPBasicApp"
**.H3.udpApp[0].destPort = 1234
**.H3.udpApp[0].messageLength = 512 bytes
**.H3.udpApp[0].sendInterval = 0.00009s
**.H3.udpApp[0].startTime = 0s
**.H3.udpApp[0].stopTime = this.startTime + 10s
**.H3.udpApp[0].destAddresses = "H4"
**.H4.numUdpApps = 1
**.H4.udpApp[0].typename = "UDPEchoApp"
**.H4.udpApp[0].localPort = 1234
[Config udp_ver]
description = "With UDP"
**.numUdpApps = 1
**.H1.udpApp[0].typename = "UDPBasicApp"
**.H1.udpApp[0].destPort = 1234
**.H1.udpApp[0].messageLength = 1024 bytes
**.H1.udpApp[0].sendInterval = 0.00009s
**.H1.udpApp[0].startTime = 0s
**.H1.udpApp[0].stopTime = this.startTime + 10s
**.H1.udpApp[0].destAddresses = "H2"
**.H2.udpApp[0].typename = "UDPEchoApp"
**.H2.udpApp[0].localPort = 1234
[General]
network = DynamicTest
tkenv-plugin-path = ../../../etc/plugins
sim-time-limit = 100s
```

```
**.eth[*].numOutputHooks = 1
**.eth[*].outputHook[0].typename = "ThruputMeter"
**.ospf.ospfConfig = xmldoc("ASConfig.xml")
**.H3.numTcpApps = 1
**.H3.tcpApp[0].typename = "TCPSessionApp"
**.H3.tcpApp[0].active = true
**.H3.tcpApp[0].connectAddress = "H4"
**.H3.tcpApp[0].connectPort = 10021
**.H3.tcpApp[0].tOpen = 0s
**.H3.tcpApp[0].tSend = 0s
**.H3.tcpApp[0].sendBytes = 50 MiB
**.H3.tcpApp[0].tClose = 0s
**.H4.numTcpApps = 1
**.H4.tcpApp[0].typename = "TCPEchoApp"
**.H4.tcpApp[0].localPort = 10021
```

ПРИЛОЖЕНИЕ И

Содержимое файла ASConfig.xml, описывающего маршрутизацию модели тестовой сети «дерево с 6 маршрутизаторами и борьбой за канал»

```
<?xml version="1.0"?>
<OSPFASConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="OSPF.xsd">
 <!-- Areas -->
 <Area id="0.0.0.0">
  <AddressRange address="RA>RB" mask="RA>RB" status="Advertise" />
  <AddressRange address="RA>R1" mask="RA>R1" status="Advertise" />
  <AddressRange address="RB>R2" mask="RB>R2" status="Advertise" />
  <AddressRange address="RA>R3" mask="RA>R3" status="Advertise" />
  <AddressRange address="RB>R4" mask="RB>R4" status="Advertise" />
 </Area>
 <Area id="0.0.0.1">
  <AddressRange address="H1" mask="H1" status="Advertise" />
 </Area>
 <Area id="0.0.0.2">
  <AddressRange address="H2" mask="H2" status="Advertise" />
 </Area>
 <Area id="0.0.0.3">
  <AddressRange address="H3" mask="H3" status="Advertise" />
 </Area>
 <Area id="0.0.0.4">
```

```
            <AddressRange address="H4" mask="H4" status="Advertise" />
        </Area>
        <!-- Routers -->
        <Router name="R1" RFC1583Compatible="true">
<BroadcastInterface ifName="eth0" areaID="0.0.0.1" interfaceOutputCost="1" />
<PointToPointInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="1"
/>
        </Router>
        <Router name="R2" RFC1583Compatible="true">
<BroadcastInterface ifName="eth0" areaID="0.0.0.2" interfaceOutputCost="1" />
<PointToPointInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="1"
/>
        </Router>
        <Router name="RA" RFC1583Compatible="true">
<PointToPointInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="1"
/>
<PointToPointInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="1"
/>
<PointToPointInterface ifName="eth2" areaID="0.0.0.0" interfaceOutputCost="1"
/>
        </Router>
        <Router name="RB" RFC1583Compatible="true">
<PointToPointInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="1"
/>
<PointToPointInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="1"
/>
<PointToPointInterface ifName="eth2" areaID="0.0.0.0" interfaceOutputCost="1"
/>
        </Router>
        <Router name="R3" RFC1583Compatible="true">
```

```
            <BroadcastInterface ifName="eth0" areaID="0.0.0.3"
interfaceOutputCost="1" />
            <PointToPointInterface ifName="eth1" areaID="0.0.0.0"
interfaceOutputCost="1" />
          </Router>
          <Router name="R4" RFC1583Compatible="true">
            <BroadcastInterface ifName="eth0" areaID="0.0.0.4"
interfaceOutputCost="1" />
            <PointToPointInterface ifName="eth1" areaID="0.0.0.0"
interfaceOutputCost="1" />
          </Router>
        </OSPFASConfig>
```

ПРИЛОЖЕНИЕ К

Содержимое файла Test.ned, описывающего топологию модели приближенной к реальным сетям

```
package inet.ospfv2.backbone;
import inet.networklayer.configurator.ipv4.IPv4NetworkConfigurator;
import inet.node.ethernet.Eth1G;
import inet.node.ethernet.EtherSwitch;
import inet.node.inet.StandardHost;
import inet.node.ospfv2.OSPFRouter;
network Backbone
{
    parameters:
        @display("p=10,10;b=736,568;bgb=702,666");
    types:
        channel C extends ThruputMeteringChannel
        {
            datarate = 100Mbps;
            thruputDisplayFormat = "#N";
        }
    submodules:
        R1: OSPFRouter {
            parameters:
                @display("p=122,272");
            gates:
                ethg[3];
        }
        N1: EtherSwitch {
            parameters:
                @display("p=204,272");
```

```
    gates:
        ethg[3];
}
N2: EtherSwitch {
    parameters:
        @display("p=122,428");
    gates:
        ethg[4];
}
R2: OSPFRouter {
    parameters:
        @display("p=204,195");
    gates:
        ethg[2];
}
R4: OSPFRouter {
    parameters:
        @display("p=47,476");
    gates:
        ethg[2];
}
R3: OSPFRouter {
    parameters:
        @display("p=122,515");
    gates:
        ethg[2];
}
N4: EtherSwitch {
    parameters:
        @display("p=204,119");
```

```
    gates:
        ethg[4];
}
N5: EtherSwitch {
    parameters:
        @display("p=501,515");
    gates:
        ethg[6];
}
R5: OSPFRouter {
    parameters:
        @display("p=372,300");
    gates:
        ethg[2];
}
R7: OSPFRouter {
    parameters:
        @display("p=523,119");
    gates:
        ethg[2];
}
R6: OSPFRouter {
    parameters:
        @display("p=372,476");
    gates:
        ethg[2];
}
N6: EtherSwitch {
    parameters:
        @display("p=286,476");
```

```
    gates:
        ethg[4];
}
N3: EtherSwitch {
    parameters:
        @display("p=47,205");
    gates:
        ethg[4];
}
H1: StandardHost {
    parameters:
        @display("p=182,355;i=device/laptop");
    gates:
        ethg[1];
}
H2: StandardHost {
    parameters:
        @display("p=204,26;i=device/laptop");
    gates:
        ethg[1];
}
configurator: IPv4NetworkConfigurator {
    parameters:
        config = xml("<config>"+
"<interface hosts='R2' towards='N1 R1' address='192.168.1.1'
netmask='255.255.255.0' />"+
"<interface hosts='H9' address='192.168.1.3' netmask='255.255.255.0' />"+
"<interface hosts='R1' towards='N1 R2' address='192.168.1.2'
netmask='255.255.255.0' />"+
```

```
"<interface hosts='R3' towards='N2 R1' address='192.168.2.1'
netmask='255.255.255.0' />"+
"<interface hosts='H1' address='192.168.2.2' netmask='255.255.255.0' />"+
"<interface hosts='H10' address='192.168.2.3' netmask='255.255.255.0' />"+
"<interface hosts='R1' towards='N2 R3' address='192.168.2.4'
netmask='255.255.255.0' />"+
"<interface hosts='R1' towards='N3 R4' address='192.168.3.1'
netmask='255.255.255.0' />"+
"<interface hosts='H4' address='192.168.3.3' netmask='255.255.255.0' />"+
"<interface hosts='H5' address='192.168.3.4' netmask='255.255.255.0' />"+
"<interface hosts='R4' towards='N3 R1' address='192.168.3.2'
netmask='255.255.255.0' />"+
"<interface hosts='H2' address='192.168.4.1' netmask='255.255.255.0' />"+
"<interface hosts='R5' towards='N4 H2' address='192.168.4.2'
netmask='255.255.255.0' />"+
"<interface hosts='R2' towards='N4 H2' address='192.168.4.3'
netmask='255.255.255.0' />"+
"<interface hosts='R7' towards='N4 H2' address='192.168.4.4'
netmask='255.255.255.0' />"+
"<interface hosts='R6' towards='N5 R5' address='192.168.5.1'
netmask='255.255.255.0' />"+
"<interface hosts='R3' towards='N5 R5' address='192.168.5.2'
netmask='255.255.255.0' />"+
"<interface hosts='R5' towards='N5 R6' address='192.168.5.3'
netmask='255.255.255.0' />"+
"<interface hosts='H6' address='192.168.5.4' netmask='255.255.255.0' />"+
"<interface hosts='H7' address='192.168.5.5' netmask='255.255.255.0' />"+
"<interface hosts='H8' address='192.168.5.6' netmask='255.255.255.0' />"+
"<interface hosts='R7' towards='N6 R6' address='192.168.6.1'
netmask='255.255.255.0' />"+
```

```
"<interface hosts='R4' towards='N6 R6' address='192.168.6.2'
netmask='255.255.255.0' />"+
"<interface hosts='R6' towards='N6 R7' address='192.168.6.3'
netmask='255.255.255.0' />"+
"<interface hosts='H3' address='192.168.6.4' netmask='255.255.255.0' />"+
"<multicast-group hosts='R*' address='224.0.0.5 224.0.0.6' />"+
"<route hosts='H*' destination='*' netmask='0.0.0.0' interface='eth0' />"+
"<route hosts='R*' destination='224.0.0.0' netmask='240.0.0.0' interface='eth0'
/>"+
"<route hosts='R*' destination='224.0.0.0' netmask='240.0.0.0' interface='eth1'
/>"+
"<route hosts='R1' destination='224.0.0.0' netmask='240.0.0.0' interface='eth2'
/>"+
"</config>");
                addStaticRoutes = false;
                addDefaultRoutes = false;
                @display("p=272,640");
            }
            H3: StandardHost {
                parameters:
                    @display("p=247,423;i=device/laptop");
                gates:
                    ethg[1];
            }
            H4: StandardHost {
                parameters:
                    @display("p=101,139;i=device/laptop");
                gates:
                    ethg[1];
            }
```

```
H5: StandardHost {
   parameters:
      @display("p=39,139;i=device/laptop");
   gates:
      ethg[1];
}
H6: StandardHost {
   parameters:
      @display("p=514,415;i=device/laptop");
   gates:
      ethg[1];
}
H7: StandardHost {
   parameters:
      @display("p=597,458;i=device/laptop");
   gates:
      ethg[1];
}
H8: StandardHost {
   parameters:
      @display("p=597,545;i=device/laptop");
   gates:
      ethg[1];
}
H9: StandardHost {
   parameters:
      @display("p=280,272;i=device/laptop");
   gates:
      ethg[1];
}
```

```
    H10: StandardHost {
        parameters:
            @display("p=78,355;i=device/laptop");
        gates:
            ethg[1];
    }
connections:
    R1.ethg[0] <--> C <--> N1.ethg[1];
    R1.ethg[1] <--> C <--> N2.ethg[2];
    N1.ethg[0] <--> C <--> R2.ethg[1];
    R2.ethg[0] <--> C <--> N4.ethg[2];
    N4.ethg[1] <--> C <--> R5.ethg[1];
    R5.ethg[0] <--> C <--> N5.ethg[2];
    N2.ethg[0] <--> C <--> R3.ethg[1];
    R3.ethg[0] <--> C <--> N5.ethg[1];
    N5.ethg[0] <--> C <--> R6.ethg[0];
    N6.ethg[2] <--> C <--> R6.ethg[1];
    R1.ethg[2] <--> C <--> N3.ethg[0];
    N3.ethg[1] <--> C <--> R4.ethg[0];
    R4.ethg[1] <--> C <--> N6.ethg[1];
    N4.ethg[3] <--> C <--> R7.ethg[0];
    R7.ethg[1] <--> C <--> N6.ethg[0];
    N4.ethg[0] <--> C <--> H2.ethg[0];
    N2.ethg[1] <--> C <--> H1.ethg[0];
    H3.ethg[0] <--> C <--> N6.ethg[3];
    H6.ethg[0] <--> C <--> N5.ethg[3];
    H7.ethg[0] <--> C <--> N5.ethg[4];
    H8.ethg[0] <--> C <--> N5.ethg[5];
    H5.ethg[0] <--> C <--> N3.ethg[2];
    H4.ethg[0] <--> C <--> N3.ethg[3];
```

```
        H9.ethg[0] <--> C <--> N1.ethg[2];
        H10.ethg[0] <--> C <--> N2.ethg[3];
}
```

ПРИЛОЖЕНИЕ Л

Содержимое файла omnetpp.ini, описывающего функционирование модели приближенной к реальным сетям

```
[Config inet-newreno]
description = "TCP <---> TCP with Reno algorithm"
*.H*.tcpType = "TCP"
**.H1.tcpApp[0].typename = "TCPSessionApp"
**.H1.tcpApp[0].active = true
**.H1.tcpApp[0].connectAddress = "H2"
**.H1.tcpApp[0].connectPort = 10021
**.H1.tcpApp[0].tOpen = 0s
**.H1.tcpApp[0].tSend = 0.5s
**.H1.tcpApp[0].sendBytes = 50 MB
**.H1.tcpApp[0].tClose = 0s
**.H2.tcpApp[0].typename = "TCPEchoApp"
**.H2.tcpApp[0].localPort = 10021
**.tcp.tcpAlgorithmClass = "TCPReno"
**.H3.numTcpApps = 1
**.H4.numTcpApps = 1
**.H5.numTcpApps = 1
**.H6.numTcpApps = 1
**.H7.numTcpApps = 1
**.H8.numTcpApps = 1
**.H9.numTcpApps = 1
**.H10.numTcpApps = 1
**.H3.tcpApp[0].typename = "TCPSessionApp"
**.H3.tcpApp[0].active = true
```

**.H3.tcpApp[0].connectAddress = "H4"

**.H3.tcpApp[0].connectPort = 10021

**.H3.tcpApp[0].tOpen = 1s

**.H3.tcpApp[0].tSend = 0.5s

**.H3.tcpApp[0].sendBytes = 35 MB

**.H3.tcpApp[0].tClose = 0s

**.H4.tcpApp[0].typename = "TCPEchoApp"

**.H4.tcpApp[0].localPort = 10021

**.H5.tcpApp[0].typename = "TCPSessionApp"

**.H5.tcpApp[0].active = true

**.H5.tcpApp[0].connectAddress = "H6"

**.H5.tcpApp[0].connectPort = 10021

**.H5.tcpApp[0].tOpen = 2s

**.H5.tcpApp[0].tSend = 0.5s

**.H5.tcpApp[0].sendBytes = 1 MB

**.H5.tcpApp[0].tClose = 0s

**.H6.tcpApp[0].typename = "TCPEchoApp"

**.H6.tcpApp[0].localPort = 10021

**.H7.tcpApp[0].typename = "TCPSessionApp"

**.H7.tcpApp[0].active = true

**.H7.tcpApp[0].connectAddress = "H9"

**.H7.tcpApp[0].connectPort = 10021

**.H7.tcpApp[0].tOpen = 3s

**.H7.tcpApp[0].tSend = 0.5s

**.H7.tcpApp[0].sendBytes = 5 MB

**.H7.tcpApp[0].tClose = 0s

**.H9.tcpApp[0].typename = "TCPEchoApp"

**.H9.tcpApp[0].localPort = 10021

**.H8.tcpApp[0].typename = "TCPSessionApp"

**.H8.tcpApp[0].active = true

**.H8.tcpApp[0].connectAddress = "H10"
**.H8.tcpApp[0].connectPort = 10021
**.H8.tcpApp[0].tOpen = 4s
**.H8.tcpApp[0].tSend = 0.5s
**.H8.tcpApp[0].sendBytes = 10 MB
**.H8.tcpApp[0].tClose = 0s
**.H10.tcpApp[0].typename = "TCPEchoApp"
**.H10.tcpApp[0].localPort = 10021
[Config udp]
description = "With UDP"
**.H1.numUdpApps = 1
**.H2.numUdpApps = 1
**.H1.udpApp[0].typename = "UDPBasicApp"
**.H1.udpApp[0].destPort = 1234
**.H1.udpApp[0].messageLength = 1024 bytes
**.H1.udpApp[0].sendInterval = 0.000083s
**.H1.udpApp[0].startTime = 10s
**.H1.udpApp[0].stopTime = this.startTime + 10s
**.H1.udpApp[0].destAddresses = "H2"
**.H2.udpApp[0].typename = "UDPEchoApp"
**.H2.udpApp[0].localPort = 1234
[General]
description = "Backbone test"
network = Backbone
tkenv-plugin-path = ../../../etc/plugins
sim-time-limit = 600s
**.ospf.ospfConfig = xmldoc("ASConfig.xml")
**.eth[*].numOutputHooks = 1
**.eth[*].outputHook[0].typename = "ThruputMeter"

ПРИЛОЖЕНИЕ М

Содержимое файла ASConfig.xml, описывающего маршрутизацию в модели приближенной к реальным сетям

```
<?xml version="1.0"?>
<OSPFASConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="OSPF.xsd">
 <!-- Areas -->
 <Area id="0.0.0.0">
  <AddressRange address="R1>N1" mask="R1>N1/" status="Advertise" />
  <AddressRange address="R1>N2" mask="R1>N2/" status="Advertise" />
  <AddressRange address="R1>N3" mask="R1>N3/" status="Advertise" />
  <AddressRange address="R2>N4" mask="R2>N4/" status="Advertise" />
  <AddressRange address="R3>N5" mask="R3>N5/" status="Advertise" />
  <AddressRange address="R4>N6" mask="R4>N6/" status="Advertise" />
 </Area>
 <Router name="R1" RFC1583Compatible="true">
 <BroadcastInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="1" routerPriority="1" />
<BroadcastInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="1" routerPriority="1" />
<BroadcastInterface ifName="eth2" areaID="0.0.0.0" interfaceOutputCost="1" routerPriority="1" />
 </Router>
 <Router name="R2" RFC1583Compatible="true">
```

<BroadcastInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="2" routerPriority="1" />
<BroadcastInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="2" routerPriority="1" />
</Router>
<Router name="R3" RFC1583Compatible="true">
<BroadcastInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="3" routerPriority="1" />
<BroadcastInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="3" routerPriority="1" />
</Router>
<Router name="R4" RFC1583Compatible="true">
<BroadcastInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="4" routerPriority="1" />
<BroadcastInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="4" routerPriority="1" />
</Router>
<Router name="R5" RFC1583Compatible="true">
<BroadcastInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="5" routerPriority="1" />
<BroadcastInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="5" routerPriority="1" />
</Router>
<Router name="R6" RFC1583Compatible="true">
<BroadcastInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="6" routerPriority="1" />
<BroadcastInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="6" routerPriority="1" />
</Router>
<Router name="R7" RFC1583Compatible="true">

<BroadcastInterface ifName="eth0" areaID="0.0.0.0" interfaceOutputCost="7" routerPriority="1" />
<BroadcastInterface ifName="eth1" areaID="0.0.0.0" interfaceOutputCost="7" routerPriority="1" />
</Router>
</OSPFASConfig>